Technical Manual

L64324
24 x 2 Fast Ethernet
Intelligent Switch

July 2001

Preliminary

LSI LOGIC®

This document is preliminary. As such, it contains data derived from functional simulations and performance estimates. LSI Logic has not verified either the functional descriptions, or the electrical and mechanical specifications using production parts.

This document contains proprietary information of LSI Logic Corporation. The information contained herein is not to be used by or disclosed to third parties without the express written permission of an officer of LSI Logic Corporation.

Document DB14-000180-00, First Edition (July 2001)
This document describes revision A of the LSI Logic Corporation L64324 24 x 2 Fast Ethernet Intelligent Switch and will remain the official reference source for all revisions/releases of this product until rescinded by an update.

To receive product literature, visit us at http://www.lsilogic.com.

LSI Logic Corporation reserves the right to make changes to any products herein at any time without notice. LSI Logic does not assume any responsibility or liability arising out of the application or use of any product described herein, except as expressly agreed to in writing by LSI Logic; nor does the purchase or use of a product from LSI Logic convey a license under any patent rights, copyrights, trademark rights, or any other of the intellectual property rights of LSI Logic or third parties.

TRADEMARK ACKNOWLEDGMENT
The LSI Logic logo design, CoreWare, DCAM, and EtherCore, are trademarks or registered trademarks of LSI Logic Corporation. ARM is a registered trademark of ARM LTD., used under license. All other brand and product names may be trademarks of their respective companies

IF

Preface

This book is the primary reference and Technical Manual for the L64324 24 x 2 Fast Ethernet Intelligent Switch. It contains a complete functional description for the L64324 and includes complete physical and electrical specifications for the L64324.

Audience

This document assumes that you have some familiarity with microprocessors and related support devices. The people who benefit from this book are:

- Engineers and managers who are evaluating the L64324 for possible use in a system
- Engineers who are designing the L64324 into a system

Organization

- Chapter 1, Introduction
- Chapter 2, Signal Descriptions
- Chapter 3, System Memory
- Chapter 4, Buffer Management
- Chapter 5, Address Resolution
- Chapter 6, Port Trunking
- Chapter 7, LED Control
- Chapter 8, Commands
- Chapter 9, Specifications

Conventions Used in This Manual

The word *assert* means to drive a signal true or active. The word *deassert* means to drive a signal false or inactive. Signals that are active LOW end in an “n.”

Hexadecimal numbers are indicated by the prefix “0x”—for example, 0x32CF. Binary numbers are indicated by the prefix “0b”—for example, 0b0011.0010.1100.1111.

The following is a list of notational conventions used throughout this manual:

Notation	Example	Meaning and Use
courier typeface	`.nwk` file	Names of commands, files, signals, symbols, pins, parts, directories, modules, and macrocells are shown in courier typeface.
bold typeface	**fd1sp**	In a command line, keywords are shown in bold, nonitalic typeface. Enter them exactly as shown.
italics	*module*	In command lines and names, italics indicate user variables. Italicized text must be replaced with appropriate user-specified items. Enter items of the type called for, using lower case.

Contents

Figures

Tables

Chapter 1
Introduction

This chapter provides an introduction to the L64324 Fast Ethernet Intelligent Switch. It contains the following sections:

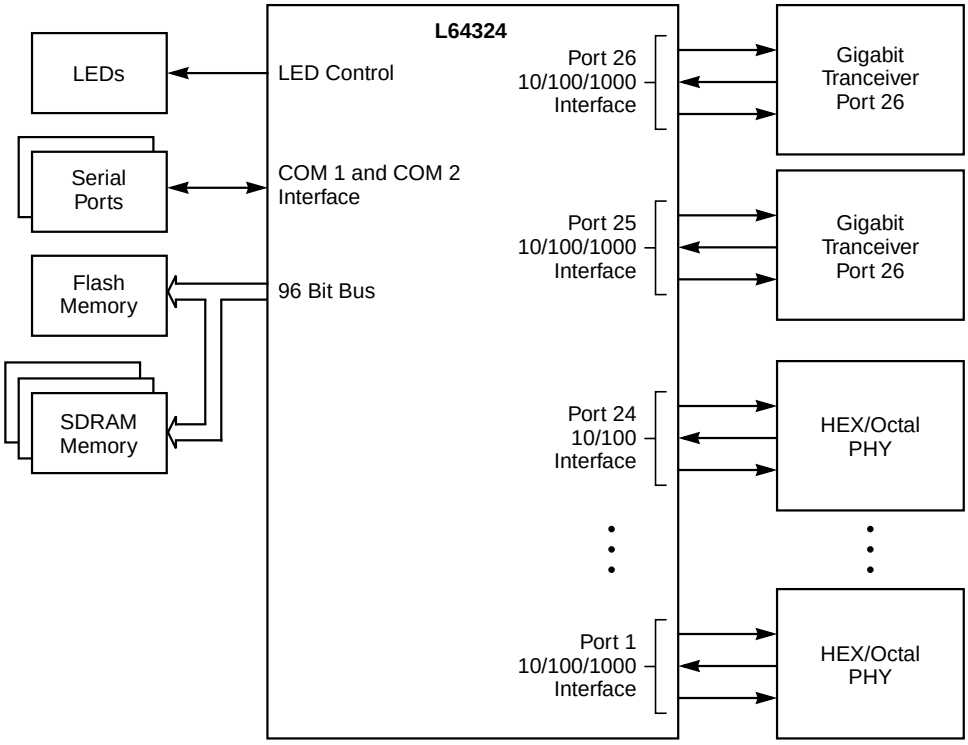
1.1 Overview

The L64324 is a highly-integrated 24-port Fast Ethernet switch with two Gigabit Transciever (uplink) ports. Each of the 24 Fast Ethernet ports operates at 10 or 100 Mbits/s, while the Gigabit Transciever ports operate at 10, 100, or 1000 Mbits/s.

The L64324 interfaces with three 2 M x 32 (64 Mbit) 125 MHz SDRAM devices, 8, 16, 32, or 64 Mbit Flash memory devices, 24 10/100 SMII Physical Layer devices (PHYs), and two 10/100/1000 Gigabit transceivers, Magnetics, RJ45 connectors, and an LED interface, to form a complete 24 x 2 Fast Ethernet switch. The L64324 also contains two integrated UART ports that allow external access for configuration and debugging.

The SDRAM is used as packet buffer storage as well as system memory. The Flash memory is used to store boot and operating system code. The LED interface drives external shift registers that control LEDs to indicate device and port operation and status. Figure 1.1 is a typical system block diagram.

Figure 1.1 Typical System Block Diagram



As shown in Figure 1.1, the L64324 has 24 10/100 Mbits/s Fast Ethernet ports as well as two 10/100/1000 Mbit/s Gigabit Transciever (uplink) ports. The uplink ports allow traffic from several lower-speed ports (1–24) to be concentrated onto one or two of the high-speed ports (25 or 26).

The L64324 allocates buffer space in system memory for packets arriving on the input ports and leaving on the output ports. The L64324 also handles switching of packet data among ports. An Available Buffer Index Table (ABIT) within the L64324 contains 4,096 index pointer locations that indicate the location of packet buffers in external SDRAM. Each entry in the table is an index to a physical memory buffer. The physical size of each buffer in external memory is 1536 bytes. Initially, the ABIT contains a single 12-bit entry for every buffer in the system. The number of entries in the table at any time depends on the number of packets buffered in the system.

A CPU Subsystem in the L64324 contains an ARM7 processor that executes instructions residing in external SDRAM memory and controls overall operation of the L64324. The CPU operates on two busses. An Advanced System Bus (ASB) is used for high-performance system modules and supports burst memory accesses and multiple bus masters. The CPU resides on the ASB. The Advanced Peripheral Bus (APB) is used for lower-performance peripherals, such as timers, GPIO, an interrupt controller, an LED port, and UART ports.

The DMA Controller interfaces to the Ethernet ports, the CPU, and the Memory Controller to allow block transfers of data between the ports and external SDRAM as well as allowing the CPU to access data and instructions from SDRAM. The DMA Controller also performs round-robin arbitration to maintain fairness among ports in accessing memory.

The Memory Controller manages external SDRAM and Flash memory devices. The SDRAM memory is used for packet storage as well as code and data space storage. The controller supports a variety of SDRAM devices. The intended memory is three 125 MHz, x 32, 64 Mbit SDRAMs. Flash memory is nonvolatile and is used for storing boot and operating system code. The controller supports 1 Mbyte, 2 Mbytes, 4 Mbytes, or 8 Mbytes Flash memory devices. The Flash and SDRAM memory devices share the Memory Controller bus; however, accesses to Flash and SDRAM are mutually exclusive.

As shown in Figure 1.1, the L64324 interfaces to external PHYs. Three octal PHYs or four hex PHYs can accommodate all 24 Fast Ethernet ports. The PHYs may then be interfaced through appropriate magnetic components and RJ45 connectors to Ethernet media. Separate Gigabit transceivers can be used to interface the two Gigabit uplink ports to the Ethernet media.

1.2 Features

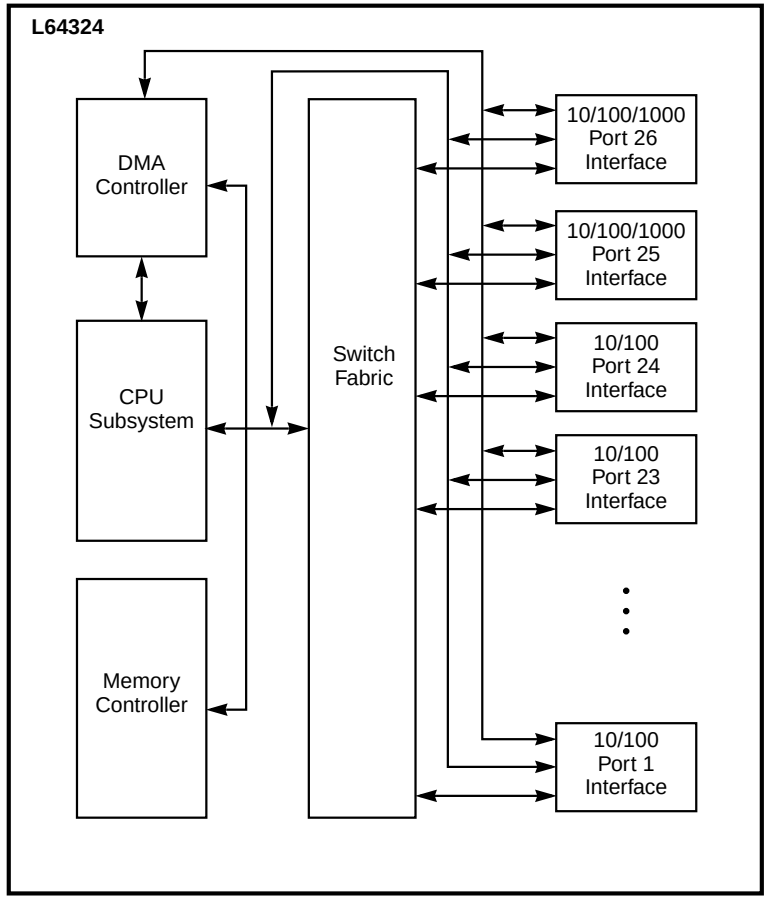
The L64324 Fast Ethernet Intelligent Switch provides the following features:

- Sustains packet transfers of over 13 Megapackets per second with 96-bit external SDRAM operating at 125 MHz.
- Capable of resolving over 4096 MAC addresses.
- Provides IEEE 802.3x flow control.
- Accommodates up to four-port trunking, supporting 800 Mbits/s.
- Supports IEEE 802.1q VLAN tagging and untagging of up to 32 VLANs.
- Supports IEEE 802.1d Spanning Tree protocol.
- Support for MII, SMII, GMII, and TBI PHY interfaces on the two 10/100/1000 Mbits/s uplink ports.
- Support for SMII PHY interface on the 24 10/100 Mbits/s Fast Ethernet ports.
- 388 PBGA package.

1.3 Functional Description

The functional description given in this section is organized around the block diagram shown in Figure 1.2. The description that follows is intended to set the stage for later chapters which explain each part of the block diagram more fully.

Figure 1.2 L64324 Block Diagram



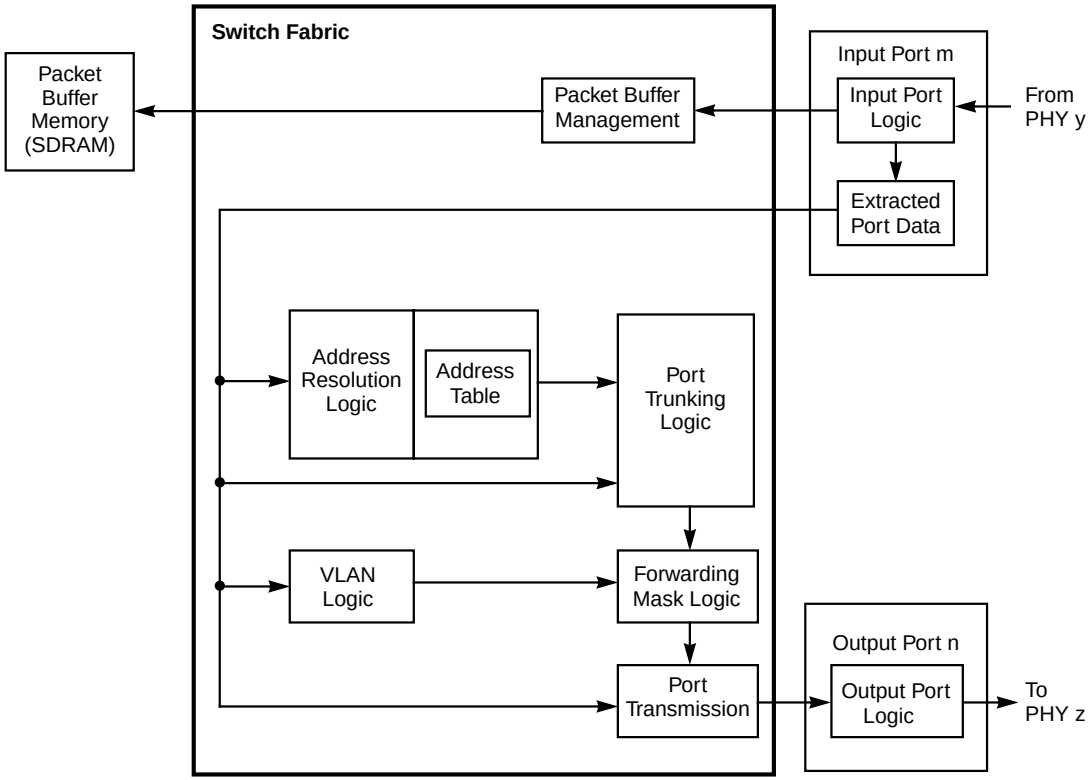
1.3.1 Switch Fabric

The heart of the L64324 is the Switch Fabric block. Under control of the CPU this block performs the following functions:

- Packet buffer management
- Address resolution
- Port Trunking
- VLAN management
- Forwarding mask
- Port transmission

A block diagram of the Switch Fabric is shown in Figure 1.3.

Figure 1.3 Switch Fabric Block Diagram



When a packet arrives at an input port, it is stored in a FIFO before the packet is transferred to packet buffer memory in 96-byte bursts. When the packet is received, the Input Port Logic requests an index. The Packet Buffer Management block maintains an Available Buffer Index Table (ABIT) that contains 4096 12-bit index entries that point to physical memory locations in the shared Packet Buffer Memory. The ABIT acts as a shared depository of available packet buffers in memory. When a port requires a buffer, an ABIT index is allocated to the port. When a packet has been transmitted, the buffer is released and made available to the system by returning the index back to the ABIT. There are 4096 entries in the ABIT. Each entry is an index to a physical memory buffer. The physical size of each buffer is 1536 bytes. Refer to Chapter 3, System Memory and Chapter 4, Buffer Management for a detailed description of packet buffer management.

As a packet is being received, the Input Port Logic parses the packet to extract data. The extracted data is used to eventually create a forwarding mask that the Port Transmission block uses to create per port transmit descriptors and a CPU receive descriptor.

Flow control can be invoked to prevent incoming packets from filling up Packet Buffer Memory or to prevent filling transmit descriptor memory on output ports. Flow control is invoked when either of the following conditions exist:

- The overall shared Packet Buffer Memory availability drops below a programmed threshold, or
- A given output port has queued the maximum programmed number of transmit descriptors and an attempt is made to add more transmit descriptors.

The Address Resolution Logic (ARL) block is responsible for the creation, maintenance, and operation of the Address Table. The Address Table contains 4096 64-byte entries that store information regarding where unicast and multicast packets are to be forwarded. The ARL is fully autonomous and does not specifically require CPU intervention. However, The CPU can read and write to all fields of the address table and is responsible for the addition and maintenance of multicast entries. Refer to Chapter 5, Address Resolution Logic for a detailed description of address resolution.

The Port Trunking block allows two or more ports to be combined to form one logical higher bandwidth link. Port trunking determines how the traffic will be sent across each of the individual links. There are several ways to make this decision. At the data link layer, this decision can be made based on the destination address, source address, or a combination of source and destination addresses. Refer to Chapter 6, Port Trunking for a detailed description of port trunking.

The VLAN Logic contains a 32-entry Content Addressable Memory (CAM). The VLAN_ID field extracted from an incoming packet or the Port VLAN ID (PVID) of the port is used to do a CAM lookup. The result of the CAM lookup is a value from 0 to 31, which in turn indexes into a VLAN table. The VLAN table contains a port mask of VLAN members and the Port Mirror, Priority, and CPU forwarding fields. The Forwarding Mask Logic Block ANDs the resulting VLAN port mask with the corresponding positions in the port mask from the Port Trunking block. Only ports that are a member of the VLAN may receive the packet.

The Port Transmission block examines the forwarding mask generated from the ARL, VLAN, Trunking, Port Mirror, and other operations to determine if a transmit descriptor must be created for a given port. Once the destination ports are known, each port is examined to determine if there are descriptor resources available to the given port. If resources are available, a descriptor is generated for each applicable output port. Each output port has a high priority and a low (default) priority transmit descriptor ring. Each transmit descriptor contains the following fields:

- A pointer to the buffer that contains the packet
- A byte count from the first byte of the destination address to the last byte of the FCS. The port logic uses this field to determine when the last byte of a packet has been read from memory.
- A processing field to determine if a VLAN and/or priority tag should be added, removed, modified or passed unaltered. The field is generated from the extracted packet data and the VLAN table.
- A Flush bit. The transmit descriptor aging logic sets this bit when an entry has been in the transmit descriptor ring longer than one second.
- An Age bit. New descriptor entries have both the Flush and Age bit reset. The transmit descriptor aging logic parses the transmit descriptor every second and sets Age bits that are “0” to “1”. The

Flush bit is set for entries found that are already set. When the port logic pulls in a descriptor from the table that has the Flush bit set, the port returns the index back to the ABIT and does not transmit the associated packet.

1.3.2 Port Interfaces

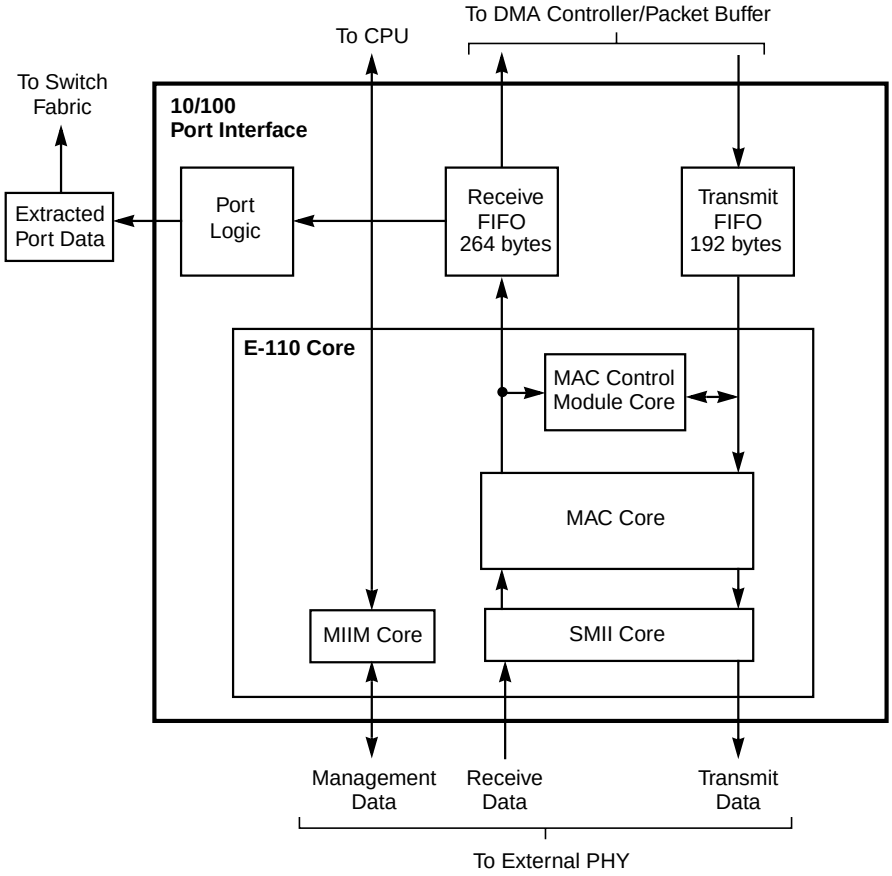
The L64324 contains two types of port interfaces:

- 10/100 Mbits/s (24 ports total)
- 10/100/1000 Mbits/s (2 ports total)

1.3.2.1 10/100 Port Interface

The L64324 contains 24 10/100 Mbits/s Port interface blocks. Each one can, under control of the CPU, transfer data at 10 or 100 Mbit/s between an external PHY and the Packet Buffer Memory. Figure 1.4 is a block diagram of a single 10/100 Port Interface.

Figure 1.4 10/100 Port Interface Block Diagram



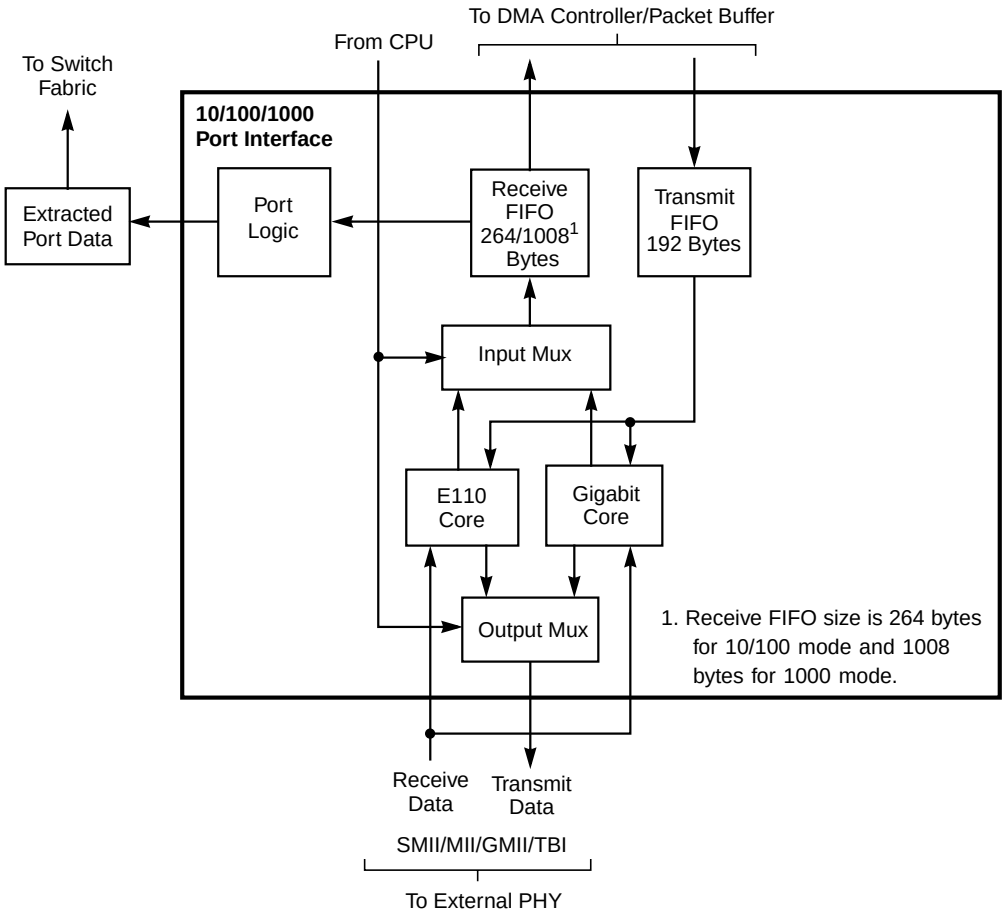
As shown in Figure 1.4, each Port Interface contains an E-110 core, which consists of the following cores:

- Media Access Control (MAC) Core: meets the requirements of the IEEE 802.3u specification and provides the ability to receive and transmit data at 10 or 100 Mbits/s.
- SMII Core: allows the port to communicate with an external PHY using a Serial MII (SMII) interface.
- MIIM Core: allows the CPU to communicate with the control and status registers within a PHY.
- MAC Control Module core: allows the host to perform the flow control functions given in the IEEE 802.3/802.3u standard.

1.3.2.2 10/100/1000 Port Interface

The L64324 contains two uplink 10/100/1000 Mbits/s Port interface blocks. Each one can, under control of the CPU, transfer data at 10, 100, or 1000 Mbits/s between an external PHY and the Packet Buffer Memory. Figure 1.5 is a block diagram of a single 10/100/1000 Port Interface.

Figure 1.5 10/100/1000 Port Interface Block Diagram



As shown in Figure 1.5, each uplink port contains an E110 core (see Figure 1.4 for details) and a Gigabit core. The uplink ports use the SMII or MII interface in 10 or 100 Mbits/s mode, and the GMII or Ten Bit Interface (TBI) in 1000 Mbits/s mode. Ports 25 and 26 can either use a 10/100 MAC or a 1000 MAC. The CPU reads the GPIO ID pins

connected to the port and selects the appropriate MAC and associated interface. For 100/1000 Mbits/s transceivers, the CPU must also read the transceiver AutoNegotiation result and select the appropriate MAC using the MAC_SELECT register.

The CPU is responsible for configuring each MAC for operation before packets can be received or transmitted. Registers exist for Interpacket Gap (IPG) control and miscellaneous configuration. For most functions, one single register is used to configure all 10/100 MACs with the same value. For other functions such as duplex operation and CSMA/CD backoff, each MAC is configured with separate values.

Each Gigabit core includes a Physical Coding Sublayer (PCS) block that can be used optionally when interfacing to fiber using the TBI. The PCS block can be bypassed when GMII is used for copper transceivers that interface to CAT5 wiring. The two uplinks can be used in any mode for aggregation of traffic to the backbone or for connectivity to local server farms.

1.3.2.3 Port FIFOS

Each Port Interface contains two FIFOs, one for receive data and one for transmit data. The Receive FIFO is 264 bytes deep for 10/100 Mbits/s mode and 1008 bytes deep for 1000 Mbits/s mode. The Transmit FIFO is 192 bytes deep for both modes. The Receive FIFO is larger because the main SDRAM arbiter gives higher priority to transmit packets than to receive packets. Incoming packet data from the Receive FIFO is sent to the Packet Buffer Memory and outgoing packet data stored in the Packet Buffer Memory is sent to the Transmit FIFO.

1.3.2.4 Port Data Extraction

As packets are received, the Port Logic block extracts the following data from the packet and forwards it to the Switch Fabric block:

- DA: 48-bit destination address.
- SA: 48-bit source address.
- Source_Port: a 5-bit value (0 to 26) equal to the number of the port that received the packet.
- PVID: the 12-bit Port VLAN ID Default value.

- PPD: the 3-bit Port Priority Default value.
- Priority: a 1-bit field that designates the internal priority of the packet.
- Priority Override: a 1-bit field if set, allows the Priority field to be overwritten by the PPD value.
- Length: 11-bit field designating the number of bytes in the packet.
- TP: Tagged Packet: 1-bit field set to 1 if a tagged packet is received.
- Null: 1-bit field is set if a tagged packet is received with a VLAN_ID of 0.
- VLAN_ID: the 12-bit VLAN_ID received in the packet.
- IGMP: a 2-bit field that indicates if the received packet is an IGMP, PIM, or OSPF protocol packet.

1.3.2.5 Forwarding Mask and Descriptors

The Switch Fabric block uses the data listed above to create a forwarding mask that is used to create per port transmit descriptors and CPU receive descriptors. A transmit descriptor contains the following information:

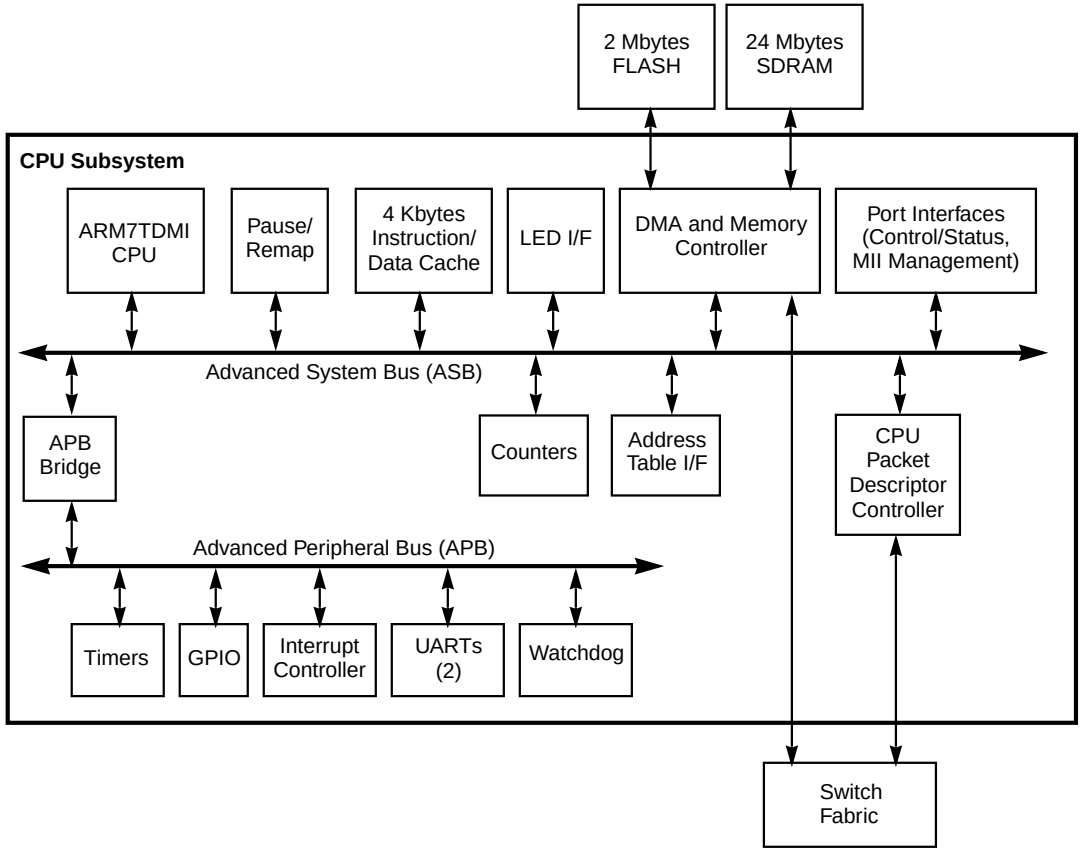
- Pointer to the buffer in Packet Buffer Memory where the packet is stored
- Packet length
- VLAN processing information
- Flush bit
- Age bit

CPU packet reception is accomplished through four separate CPU receive descriptor rings. The rings are prioritized, allowing important time-critical packets to be processed before lower priority, nonessential packets. Packets that must not be lost (Spanning Tree, IGMP, or SNMP) are placed in the highest priority queue. These packets generally do not arrive at extremely high rates. If they do, the software implementation may discard SNMP packets before discarding others, such as Spanning Tree. The lowest priority queue is used for opportunistic processing of RMON packets.

1.3.3 CPU Subsystem

The L64324 CPU Subsystem consists of an ARM7TDMI processor that implements the Advanced Microcontroller Bus Architecture (AMBA). A block diagram of the CPU subsystem is shown in Figure 1.6.

Figure 1.6 CPU Subsystem Block Diagram



As shown in Figure 1.6, the CPU Subsystem operates on two buses. The Advanced System Bus (ASB) is a high-speed, high-performance bus and the Advanced Peripherals Bus (APB) is a lower-performance bus designed for lower speed peripherals.

The major components and features of the CPU Subsystem are:

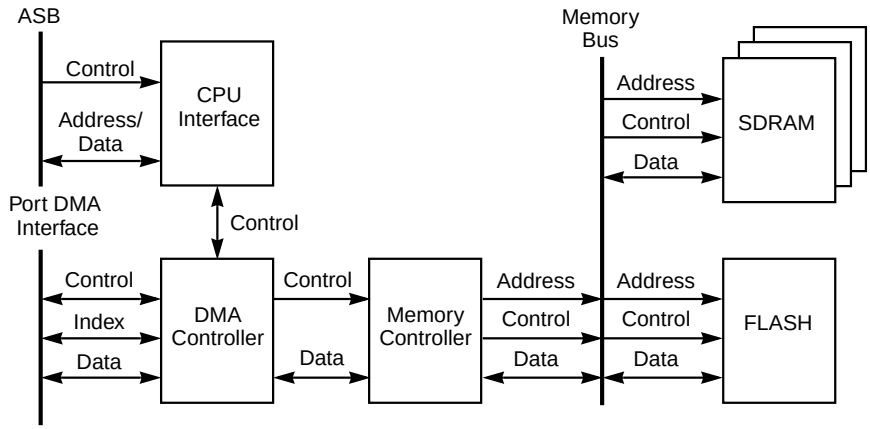
- Interrupt Controller with the following interrupt sources:
  - Timer triggers (6)
  - Watchdog
  - Bus error
  - Software programmable
  - Transmit and receive debug
  - Two UARTS
  - Switch Core
  - DMA Controller transmit and receive
  - GPIO
  - Speed sense
- 21 GPIOs for generating interrupts and for general board status
- Two independent timers
  - Independent clock prescaler for each timer
  - Programmable high/low-speed clock select for Timer 1
  - Programmable high/low-speed clock select for Timer 2
  - Six trigger point registers for generating interrupts
  - Selectable comparison with either timer
- Two 16550-compatible UARTs
  - 16-byte FIFO
  - No 5- or 6-bit character support
  - 9-bit character support
  - Separate clock from main system
  - Modem control pins (CTS, RTS, DCD, DTR, DSR, and RI) routed out on one channel only
  - Speed Sense
- Watchdog Timer
  - Interrupt on expire, followed by reset if still no response
  - Protected against inadvertent software update

- Memory Protection
- Pause and Remap Unit
  - Pause system until interrupt
  - Silicon revision/ID
  - Remap from reset map to runtime map
  - Reset status
  - Software reset

1.3.4 DMA and Memory Controllers

Because the L64324 must support 24 10/100 Mbits/s ports and two 1000 Mbits/s ports plus the CPU, a high-performance memory system is required to serve each port at the correct speed. Memory is accessed using a DMA Controller and a Memory Controller. The Memory and DMA Controllers are placed in the system as shown in Figure 1.7.

Figure 1.7 DMA Controller System Interfaces



1.3.4.1 DMA Controller

The DMA Controller is responsible for arbitrating memory requests from the CPU and Port Interfaces and interfacing to the Memory Controller to transfer data to and from either Flash or SDRAM memory. CPU instructions and data are accessed using Flash memory and packet data

is accessed using the SDRAM memory. The DMA Controller passes memory control signals and data to the Memory Controller, which then directly accesses memory.

1.3.4.2 Memory Controller

The Memory Controller provides all of the signals necessary to read, write, and refresh SDRAM on a 96-bit data width. The controller is capable of the following types of transactions: read, write, burst write, precharge, mode register set, refresh, and power down.

A refresh timer generates periodic refresh requests, while the DMA Controller provides SDRAM/Flash transaction requests as well as start address information.

The Memory Controller also provides access to the Flash memory, which is accessed on an 8-bit data bus width. Because Flash memory is much slower than SDRAM, accesses to Flash and SDRAM are mutually exclusive. The Memory Controller is capable of Flash read, erase, and program cycles, as well as programmable wait state and turnaround cycles.

Chapter 2
Signal Descriptions

This chapter describes the interfaces of the L64324 24 x 2 Fast Ethernet Intelligent Switch and the signals used in each interface. This chapter consists of the following sections:

2.1 Interfaces

Figure 2.1 shows the different interfaces. Several pins of the L64324 are dual-purpose and contain signals that are sampled at reset. Several other pins are dual-purpose and provide mode specific signals.

The signals sampled at reset are:

- FLASH_W[3:0], Flash Waits shared with DATA[71:68]
- SS[1:0], System Speed shared with DATA[67:66]
- IB_ID[3:0], Board Revision shared with DATA [83:80]
- HW_ID[3:0], Hardware Revision shared with DATA[85:84, 79:78]

- DEBUGn, Debug Mode shared with DATA[86]
- BENCHn, Bench Mode shared with DATA[87]

Dual-purpose packet buffer interface pins are:

- FADDR[22:0], Flash Memory Address shared with DATA[30:8]

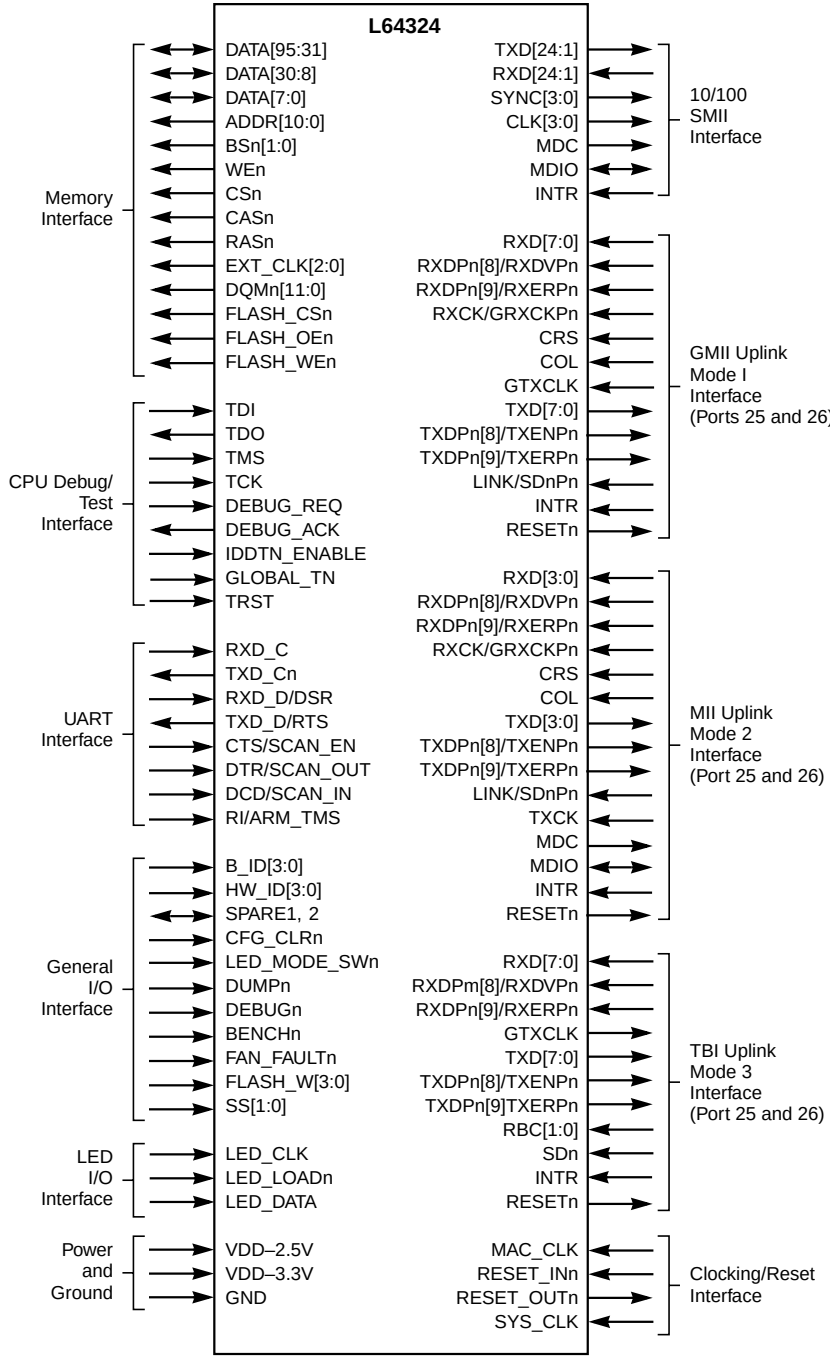
Dual-purpose pins in GMII uplink (Mode 1) and MII uplink (Mode 2) are:

- RXDV (Port 25 and 26), Receive Data Valid shared with RXD[8] (Port 25 and 26)
- RXER (Port 25 and 26), Receive Error shared with RXD[9] (Port 25 and 26)
- RXCK (Port 25 and 26), 125 MHz Receive Clock shared with GRXCK (Port 25 and 26)
- TXEN (Port 25 and 26), Transmit Enable shared with TXD[8] (Port 25 and 26)
- TXER (Port 25 and 26), Transmit Error shared with TXD[9] (Port 25 and 26)

Dual-purpose pins in TBI uplink (Mode 3) are:

- RXD[8] (Port 25 and 26), Receive Data bit [8] shared with RXDV (Port 25 and 26)
- RXD[9] (Port 25 and 26), Receive Data bit [9] shared with RXER (Port 25 and 26)
- SDn (Port 25 and 26), Signal Detect shared with LINK (Port 25 and 26)
- TXD[8] (Port 25 and 26), Transmit Data bit [8] shared with TXEN (Port 25 and 26)
- TXD[9] (Port 25 and 26), Transmit Data bit [9] shared with TXER (Port 25 and 26)

Figure 2.1 Signal Interfaces



2.2 Memory Interface

This section describes the memory interface (Packet Buffer) signals. The packet buffer is used as a store and forward main memory for the L64324. This memory region is also mapped to the ARM7 address map for additional CPU related storage as well as diagnostics.

DATA[95:31] **Packet Buffer Data** **I/O**
Upper 64 bits of SDRAM organized as big endian.

DATA[30:8]/FADDR[22:0]
Packet Buffer Data/Flash Address **I/O**
SDRAM Data [30:8]. Data is organized as Big endian. Flash Address[22:0] when FLASH_CS is asserted.

DATA[7:0]/ **Data** **I/O**
SDRAM Data[7:0] and Flash Data [7:0].

ADDR[10:0] **Packet Buffer Address** **Output**
ADDR[10:0]: is the Row Address while ADDR[7:0] is the Column Address.

BSn[1:0] **SDRAM Internal Bank Select** **Output, Active L**
Selects one of four memory banks.

WEn **Packet Buffer Write Enable** **Output, Active L**
WEn, when asserted, Write enables DATA[95:0].

CSn **SDRAM Chip Select** **Output, Active L**

CASn **SDRAM CAS** **Output, Active L**

RASn **SDRAM RAS** **Output, Active L**

EXT_CLK[2:0] **SDRAM 125 MHz Clock** **Output**

DQMn[11:0] **Output, Active L**
DQMn[0] byte selects DATA[7:0].
DQMn[1] byte selects DATA[15:8].
DQMn[2] byte selects DATA[23:16].
•
•
•

DQM[11] byte selects DATA[95:88].

During a read these signals function as output enable. During a write these signals function as write enable.

FLASH_CSn **Flash Chip Select** **Output, Active L**

FLASH_OEn **Flash Output Enable** **Output, Active L**

FLASH_WEn **Flash Write Enable** **Output, Active L**

2.3 CPU Debug and Test Interface

TDI **Scan Chain Input** **Input**
TDI is the input to the JTAG scan chain.

TDO **Scan Chain Output** **Output**
TDO is the output of the JTAG scan chain l

TMS **Test Mode Select** **Input**
Core JTAG IEEE 1149.1 signal

TCK **Test Clock** **Input**
Core JTAG IEEE 1149.1 signal

DEBUG_REQ **Enter Debug Mode** **Input, Active H,**
The DEBUG_REQ signal is asserted to quickly enter debug mode after receiving a “Break In” signal from the logic analyzer.

DEBUG_ACK **Debug Mode Entry/Exit Detect** **Output, Active H**
The DEBUG_ACK signal, when asserted indicates entry or exit from debug mode.

IDDTN_ENABLE
IDDTN Ring Enable **Input**
When HIGH, IDDTN_ENABLE enables all pull-up/pull-down structures.

GLOBAL_TN **Input, Active H**
Controls TN ring of the chip. This pin should be a no connect.

TRST **Test Reset** **Input**
TRST signal when asserted is the JTAG reset signal.

2.4 10/100 SMII Interface

TXD[24:1] **SMII Transmit data** **Output**
TXD[24:1] provides the per-port SMII transmit data output for ports 1–24 of the L64324. These signals interface to 10/100 Physical Layer devices.

RXD[24:1] **SMII Receive control and data** **Input**
RXD[24:1] provides the per-port SMII receive data input for ports 1–24 of the L64324. These signals interface to 10/100 Physical Layer devices.

SYNC[3:0] **SMII Synchronization** **Output**
When asserted, SYNC defines the first bit of the 10-bit SMII segment word.

SYNC[0] defines the segment start for ports 1, 2, 3, 7, 8, and 9. SYNC[1] defines the segment start for ports 4, 5, 6, 10, 11, and 12. SYNC[2] defines the segment start for ports 13, 14, 15, 19, 20, and 21. SYNC[3] defines the segment start for ports 16, 17, 18, 22, 23, and 24.

CLK [3:0] **SMII Receive and Transmit Clock** **Output**
Receive and transmit timing is relative to this clock.

CLK[0] provides timing for ports 1, 2, 3, 7, 8, and 9. CLK[1] provides timing for ports 4, 5, 6, 10, 11, and 12. CLK[2] provides timing for ports 13, 14, 15, 19, 20, and 21. CLK[3] provides timing for ports 16, 17, 18, 22, 23, and 24.

MDC **MII Management 2.5 MHz clock** **Output**
MDC is a 2.5 MHz clock for timing PHY IntMII management data.

MDIO **MII Management Data** **Bidirectional**
MDIO is used to access PHY management registers. MDIO is used to address all 26 PHYs connected to the L64324. This interface operates at 2.5 MHz or 25 MHz under software control.

INTR **Interrupt** **Input**
INTR is a global interrupt for SMII ports 1–24.

2.5 Gigabit Uplink Modes

The uplink ports refer to the two Gbit ports (Port 25 and 26) that can be run in either 10/100/1000 Mbits/s. Both ports can operate in one of the four different interface modes: SMII, MII, GMII, and Ten Bit Interface (TBI). The mode is selected using the MAC_SELECT register. Each uplink port can be in any of the modes independent of the other uplink port.

Note that the following sections describe the pin functions for only three different modes: **GMII Uplink (Mode 1), MII Uplink (Mode 2),** and **TBI Uplink (Mode 3)**. The pin functions for the fourth mode, SMII, were previously described. The signal I/O for the uplinks amoung the three modes of operation overlap and are described below.

2.5.1 GMII Uplink Interface (Mode 1)

The Gigabit MII (GMII) interface is designed to comply with the IEEE 802.3ab/dx.x specification. Each port operating in GMII mode has an identical set of signals listed as follows:

RXD[7:0] **Receive Data (Port 25 and 26)** **Input**
Receive data is transferred from the PHY to the MAC synchronously with respect to RXCK. These signals are valid when RXDV is asserted

RXDV **Receive Data Valid (Port 25 and 26)** **Input**
Receive data valid is transferred from the PHY to the MAC synchronously with respect to RXCK.

RXER **Receive Error (Port 25 and 26)** **Input**
When a detectable error such as a symbol error is detected, RXER is synchronously asserted with respect to RXCK for one or more cycles. RXER is driven by the PHY.

GRXCK **Receive Clock (Port 25 and 26)** **Input**
GRXCK is a continuous 125 MHz Receive clock sourced by the PHY and used as the timing reference for RXD, RXDV, and RXER. The duty cycle is 35/65. There are times when the PHY may stretch the HIGH or LOW clock period. There is no assumed phase relationship between GRXCK and GTXCLK.

CRS **Carrier Sense (Port 25 and 26)** **Input**
CRS is carrier sense from the 1000BASE-T PHY. CRS is asserted when the PHY detects that the receive or transmit medium is not idle. This signal is used in the half duplex mode and need not be synchronous.

COL **Collision (Port 25 and 26)** **Input**
Collision from the 1000BASE-T PHY. Asserted by the PHY for the duration of a collision on the medium. This signal is used in Full Duplex mode only and need not be synchronous.

GTXCLK **Transmit Clock (Port 25 and 26)** **Input**
Continuous 125 MHz transmit clock generated by the PHY. TXD, TXEN, and TXER are generated by the MAC synchronously with respect to GTXCLK.

TXD[7:0] **Transmit Data (Port 25 and 26)** **Output**
Transmit data stream from the GMAC to the 1000BASE-T PHY. Data is transmitted synchronously to GTXCLK.

TXEN **Transmit Data Enable (Port 25 and 26)** **Output**
Transmit data valid to the 1000BASE-T PHY. TXEN is asserted when valid data is driven onto TXD[7:0]. TXEN is active HIGH and is asserted synchronously with respect to GTXCLK.

TXER
Transmit Error (Port25 and 26) **Output**
Transmit error and carrier extension indication to the 1000BASE-T PHY. When TXER is asserted, the PHY does not transmit the data presented on the TXD lines.

LINK **Link (Port 25 and 26)** **Input**
This signal is available from GPIO 3.

INTR **Interrupt** **Input**
INTR is a global interrupt for GMII ports 25 and 26.

RESETn **Uplink (Port 25 and 26)** **Output**
This output directly controls the reset of the uplink module and is controlled using the MAC_SEL register.

2.5.2 MII Uplink Interface (Mode 2)

RXD[3:0] **Receive Data (Port 25 and 26)** **Input**
Receive data is transferred from the PHY to the MAC synchronously with respect to RXCK. RXD[3:0] are valid when RXDV is asserted.

RXDV **Receive Data Valid (Port 25 and 26)** **Input**
Receive data valid is transferred from the PHY to the MAC synchronously with respect to RXCK. RXD[3:0] are valid when RXDV is asserted.

RXER **Receive Error (Port 25 and 26)** **Input**
When a detectable error such as a symbol error is detected, RXER is synchronously asserted with respect to RXCK for one or more cycles. RXER is driven by the PHY.

RXCK **Receive Clock (Port 25 and 26)** **Input**
RXCK is a continuous 25 MHz receive clock sourced by the PHY and used as the timing reference for RXD, RXDV, and RXER. The duty cycle is 35/65.

CRS **Carrier Sense (Port 25 and 26)** **Input**
CRS is carrier sense from the 1000BASE-T PHY. CRS is asserted when the PHY detects that the receive or transmit medium is not idle. This signal is used in the half duplex mode and need not be synchronous. CRS remains asserted during collisions.

COL **Collision (Port 25 and 26)** **Input**
COL is asserted HIGH by the PHY for the duration of a collision on the medium. This signal need not be synchronous. This signal is not used in full-duplex mode.

TXD[3:0] **Transmit Data (Port 25 and 26)** **Output**
TXD[3:0] is the MII transmit data from the GMAC to the PHY. Data is transmitted synchronously to TXCK. TXD[3:0] is valid only when TXEN is asserted.

TXEN **Transmit Enable (Port 25 and 26)** **Output**
Transmit data valid to the PHY. TXEN is asserted when valid data is driven onto TXD[3:0]. TXEN is active HIGH and is asserted synchronously with respect to TXCK.

TXER **Transmit Error (Port 25 and 26)** **Output**
Transmit Error to the PHY. When TXER is asserted, the PHY does not transmit data presented on the TXD[3:0] lines.

LINK **Link (Port 25 and 26)** **Input**
This signal is available from GPIO 3.

TXCK **Transmit Clock (Port 25 and 26)** **Input**
TXCK is a continuous 25 MHz transmit clock generated by the PHY. TXD, TXEN, and TX_ER are generated by the MAC synchronously with respect to TXCK.

MDC **Management 2.5 MHz Clock (Port 25 and 26)** **Output**
MDC is the management 2.5 MHz signal generated by the MAC and sent to the PHY.

MDIO **Management Data (Port 25 and 26)** **Input/Output**
This signal is used to transfer a serial bit stream synchronously with MDC, between the management entity and the PHYs. The serial stream protocol is defined in the IEEE 802.3u standard.

INTR **Interrupt** **Input**
INTR is a global interrupt for MII ports 25 and 26.

RESETn **Uplink Module Reset (Port 25 and 26)** **Output**
This output directly controls the reset of the uplink module and is controlled from the MAC_SEL register.

2.5.3 TBI Uplink Interface (Mode 3)

The Gbit network 10-bit interface is designed to comply with the IEEE 802.3z/dx.x specification.

RXD[7:0] **Receive Data (Port 25 and 26)** **Input**
Receive data is transferred from the PHY to the MAC synchronously with respect to RBC[1:0].

RXD[8] **Receive Data Bit 8 (Port 25 and 26)** **Input**
RXD[8] of the receive data transferred from the PHY to the MAC.

RXD[9] **Receive Data Bit 9 (Port 25 and 26)** **Input**
RXD[9] of the receive data transferred from the PHY to the MAC.

GTXCLK **Transmit Clock (Port 25 and 26)** **Intput**
GTXCLK is a continuous 125 MHz transmit clock generated by the PHY. TXD, TXEN, and TXER are generated by the MAC synchronously with respect to GTXCLK.

TXD[7:0] **Transmit Data (Port 25 and 26)** **Output**
TXD[7:0] is the Transmit data stream from the GMAC to the PHY. Data is transmitted synchronously to GTXCLK. TXD[7:0] is valid only when TXEN is asserted.

TXD[8] **Transmit Data Bit 8 (Port 25 and 26** **Output**
TXD[8] of the transmit data from the GMAC to the PHY.

TXD[9] **Transmit Data Bit 9 (Port 25 and 26)** **Output**
TXD[9] of the transmit data from the GMAC to the PHY.

RBC[1:0] **Receive Clock** **Input**
RBC[0] is a 62.5 MHz clock, Phase 0. RBC[1] is a 62.5 MHz clock, Phase 180.

SDn **Gbit Signal Detect (Port 25 and 26)** **Input**
This signal is asserted (3.3 V) when a valid signal is detected. This signal is not part of the TBI standards recommendation.

INTR **Interrupt** **Input**
INTR is a global interrupt for TBI ports 25 and 26.

RESETn **Uplink Module Reset (Port 25 and 26)** **Output**
This output directly controls the reset of the uplink module and is controlled from the MAC_SEL register.

2.6 Clocking/Reset Interface

MAC_CLK **125 MHz Clock Oscillator** **Input**
This signal is the timing source for the MAC and PHY interface timing. The oscillator should have a duty cycle tighter than 40/60.

RESET_INn **Active Low Reset In** **Input**
The system global active LOW reset is connected to this input.

RESET_OUTn Active Low Reset **Output**
This signal is driven by the OR of the GL_RESET and Watchdog_Reset internal signals. It has an open-drain output and should pull-up to 3.3 V.

SYS_CLK **System Clock** **Input**
SYS_CLK 125 MHz system clock is driven from an external clock oscillator.

2.7 UART Interface

RXD_C **RXD Data for Product Console** **Input**

TXD_Cn **TXD Data for Product Console** **Output**

RXD_D/DSR **Data Set Ready** **Input**
Data Set Ready - Normal Operation.

RXD_D - when in debug mode.

TXD_D/RTS **Request To Send** **Output**
Request To Send - Normal Operation.

TXD_D - when in debug mode.

CTS/SCAN_EN
Scan Enable **Input**
During test this pin supplies SCAN_ENABLE input.

DTR/SCAN_OUT **Input**
During test this pin supplies SCAN_OUT data.

DCD/SCAN_IN **Input**
During test this pin supplies SCAN_IN data and control.

RI/ARM_TMS **Input**
ARM_TMS signal used to run scan against the ARM7. The ARM TAP will be sharing the TDI, TCK and TDO pins with the main CPU debug TAP. If this signal is not pulled up to +3.3 V the device cannot enter the debugging mode.

2.8 LED Interface

LED_CLK **Output**
LED_Clock (122.07 kHz).

LED_LOADn **Output, Active L**
Asserted to Load Latches with LED data bytes.

LED_DATA **Output**
64-bit LED data stream.

2.9 General I/O Interface

B_ID[3:0] **Board Revision of a Given Product or Hardware ID** **Input, Pull-Up**

HW_ID[3:0] **Hardware ID or Product ID** **Input, Pull-Up**

SPARE 1, 2 **General Purpose I/O for CPU** **I/O, Pull-Up**
SPARE 1 is used as RTCK for the ARM core. Only SPARE 2 is available.

CFG_CLRn **Input, Active L, Pull-Up**
Connected to configuration switch, should be debounced.

LED_Mode_SWn **Input, Active L, Pull-Up**
Input directly connected to LED Mode Momentary Switch.

DUMPn **Input, Active L, Pull-Up**
Used to trigger software dump.

DEBUGn **Input, Active L, Pull-Up**
Sampled at reset to determine debug mode.

BENCHn **Input, Active L, Pull-Up**
Sampled at reset.

FAN_FAULTn **Input, Active L, Pull-Up**
When sampled LOW, the FAN_FAULT LED is lit.

FLASH_W[3:0] **Input, Pull-Up**
Sampled on DATA[71:68] at reset.

SS[1:0]	**System Frequency Indicator** Sampled on DATA[86, 87] on reset. Indicates system frequency, i.e. 125, 133, 143 MHz.	**Input, Pull-down**

2.10 Power and Ground Pins

GND	**Logic Ground** Internal core logic ground reference.	**Input**
VDD-2.5V	**Logic Power** Internal core logic 2.5 V supply voltage.	**Input**
VDD-3.3V	**Logic Power** I/O 3.3 V supply voltage.	**Input**

Chapter 3
System Memory

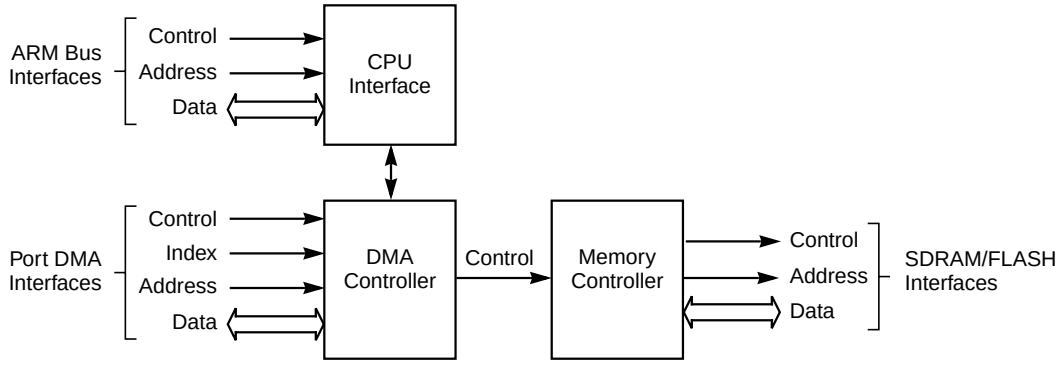
This chapter describes the system memory of the L64324 24 x 2 Fast Ethernet Intelligent Switch and consists of the following sections:

- Section 3.1, “Memory Control System”
- Section 3.2, “DMA Control”
- Section 3.3, “Memory Control”

3.1 Memory Control System

The L64324 Switch has a shared memory system consisting of SDRAM and Flash memory. The memory space is divided into three parts, packet (buffer) memory space for packet storage, CPU memory space for code/data storage, and Flash memory space for boot and operating system code. The Flash and CPU spaces are overlapped. The system memory supports the transmit and receive operation of 24 10/100 and two 10/100/1000 Ethernet ports. This high performance memory system is required to provide adequate speed. The memory system is also responsible for loading the boot code from the Flash memory to SDRAM. Figure 3.1 is a block diagram of the L64324 Memory Control System.

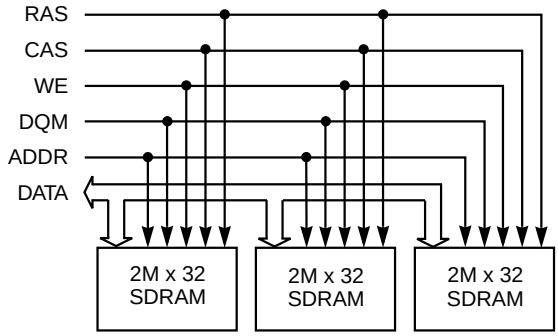
Figure 3.1 Memory Control Block Diagram



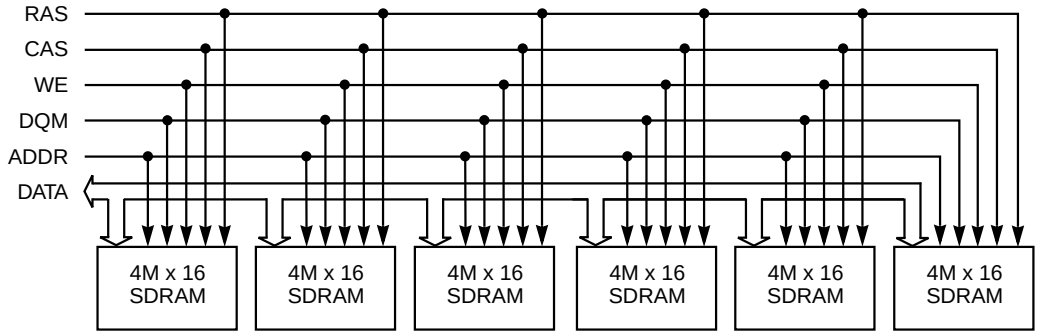
3.1.1 SDRAM Memory Configurations

The memory control system supports up to three 2M x 32, 125 MHz, 64 Mbit SDRAMs. Figure 3.2 shows examples of different SDRAM configurations. The SDRAM/Flash controller implements the 8-bit Flash memory interface protocol. Flash memory size can vary from 2M x 8 to 8M x 8.

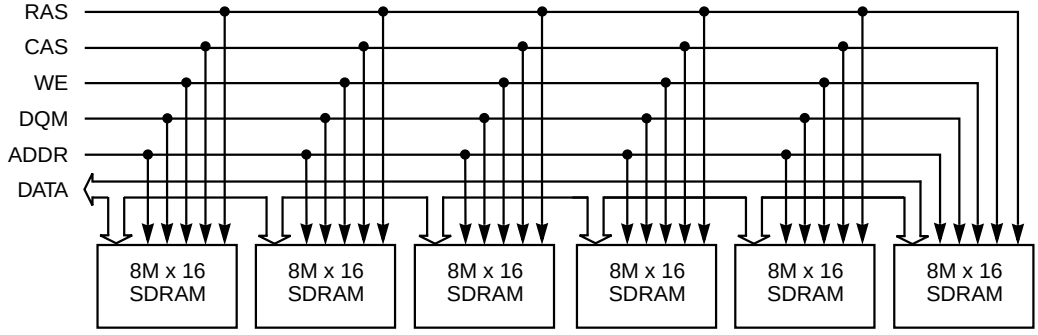
Figure 3.2 SDRAM Configuration Examples



64 Mbits, 2M x 32 Chips, 96-Bit Bus (Preferred)



64 Mbits, 4M x 16 Chips, 96-Bit Bus



128 Mbits, 8M x 16 Chips, 96-Bit Bus

3.1.2 Address Mapping

The L64324 SDRAM memory has two major spaces, Packet Buffer memory space and CPU memory space. Table 3.1 lists the physical start and end addresses of these memory spaces.

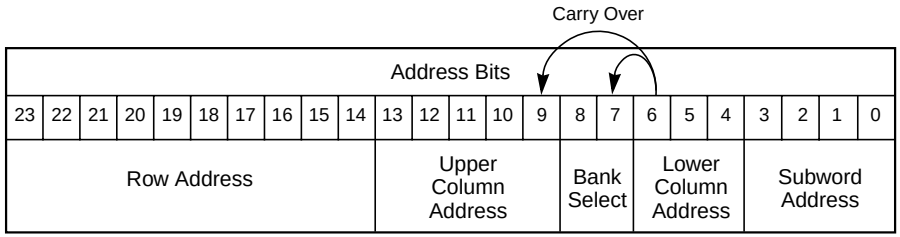
Table 3.1 96-Bit Bus Memory Map

Memory Space	Physical Start Address	Physical End Address
CPU Memory Space (6 Mbyte)	0x000000	0x5.FFFF
Packet Buffer Memory (18 Mbyte)	0x600000	0x17.FFFF

3.1.3 Packet Memory Addressing

The greatest memory efficiency and packet throughput are gained when consecutive accesses occur on different banks within the SDRAM. To maximize the bandwidth, a memory addressing method is used so that each packet uses all banks. As shown in Figure 3.3, the spiral addressing requires that the lower column address carry the bank select bits and higher column address increment at the same time. In both cases, the 7 and 6 LSB column address is module -128/64. To compensate for a minimum size packet for bank 0, the starting bank is a function of the least significant bits of the index address. A fixed packet buffer size of 1536 bytes per packet for a 16-bit bus is used for all three configurations.

Figure 3.3 96-Bit Bus Packet Memory Address Space



In the 96-bit bus configuration, a row contains 4 banks x 256 columns x 96-bit word for a total of 12288 bytes. From the maximum packet size, 1522 bytes, 12288/1522 = 8.07, 8 maximum size packets can fit in a

given row. So the 64 bit bus packet buffer size is 12288/8 = 1536 bytes. Each buffer does contain an even number of the 96 byte blocks. 1536/96 = 16, so, there are 16 full blocks in a buffer. The 7 LSB column address will do module -128 increment. The start block is unique for each packet.

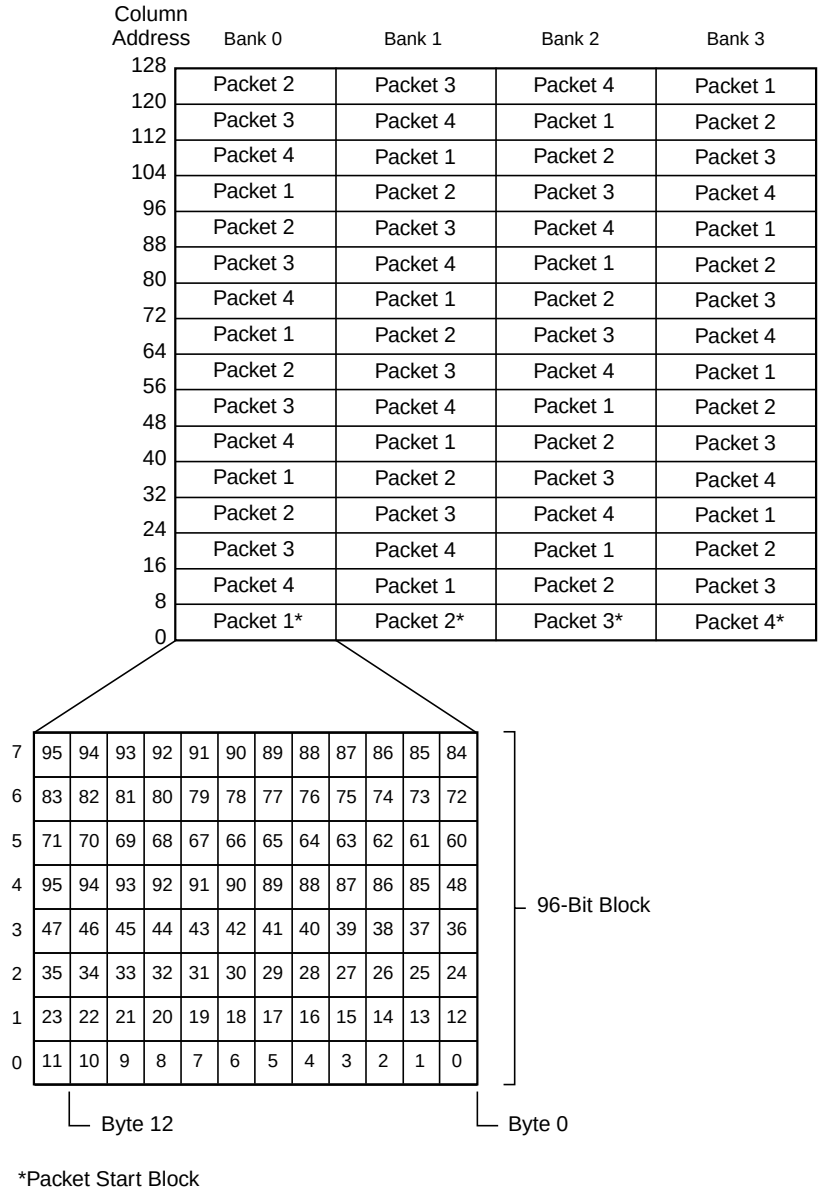
The packet memory addressing scheme has these features:

- Rows: 512
- Columns: 256
- Block size: 96 bytes
- Packet buffer size: 1536 bytes, (16 blocks)
- Packet buffers per row: 8

It is important that port logic never allow more than 1536 bytes to be transferred, otherwise the row could wrap around or cross over into the next packet and destroy the data stored there.

As memory fills, the column address increments from 0 to 256 before cycling back to 0 and repeating. For each 8 columns transferred, the bank toggles up one and to the next consecutive bank. This results in the spiral effect shown in Figure 3.4. For example, if the CPU is reading memory sequentially, the blocks associated with packet 1 in Figure 3.4 would be accessed first, then the bytes from packet 2, followed by the bytes from packet 3.

Figure 3.4 96-Bit Bus Spiral Addressing



*Packet Start Block

3.1.4 CPU Memory Addressing

The CPU memory space is arranged the same as packet memory space. The CPU and packet memory are byte addressable. Hardware hides the spiral structure and presents a linear space to the CPU by means of address translation. The CPU does the index to (logical) byte address translation. Hardware does the (logical) byte address to physical RAS, CAS, and address translation. This hardware translation is located in the DMA module to provide a more generic SDRAM controller. The address translation logic is also responsible for breaking any single burst access that crosses the block boundary into separate bursts within different blocks. This allows the hardware to hide the spiral or block memory structure from the CPU and port logic.

3.1.5 Address Translation

Two agents access the packer buffer, port logic and the CPU. Port Logic does sequential access, while the CPU does random byte access. The CPU does index/descriptor to byte address translation and DMA does the physical RAS, CAS, and bank address translation. For Port logic, the DMA does index to byte address translation, and byte address to physical RAS, CAS, and bank address translation. The Address Translation logic is also responsible for breaking any single burst access that crosses the block boundary into separate bursts to different blocks. In this way, the hardware hides the spiral or block memory structure from the CPU or port logic.

Index to physical address translation:

- Row Address = Index divided by 8
- Bank Select = The second and third LSBs of the index
- Column Address = The first LSB of the index determines if the column starts at 0x00 or 0x80

Index to byte address translation:

- Byte address: = Row * (0x3000) + Bank Select [1:0] * (0x600) + Column [7] * (0x1800) + Column [6:0] * (0xc)

Byte Address to Physical Address Translation:

- Row Address = Byte address divided by row size 12288(0x3000), (A/0x3000).
- Row Address[8:0] = (A/0x3000)

Bank Select:

- Bank select is equal to the number of blocks in the current packet plus start bank offset, and module 4
- Number of blocks in the current packet is equal to remainder of the byte address divided by packet buffer size, 1536(0x600), then divided by block size 96(0x60)
- Start bank, which is quotient of the byte address divided by packet buffer size, 1536(0x600)
- Bank Select[1:0] = (A – ((A/0x600) * 0x600))/0x60 + (A/0x600)

Column Address:

- Column address is equal to the number of words in the current packet plus the start column.
- Number of words in the current packet is equal to remainder of the byte address divided by packet buffer size, 1536(0x600), divided by the word size 12(0xc), take the seven LSBs.
- Start column is either 0x00 or 0x80 and is decided by the LSB of the quotient from the byte address divided by packet buffer size 1536(0x600).
- Column[7:0] = (A – (A/0x600) * 0x600)/0xc + [(A/0x600) + 0x01] * 0x80

Table 3.2 lists examples of byte index to byte address and byte address to physical address.

Table 3.2 Index to Byte Address and Byte Address to Physical Address

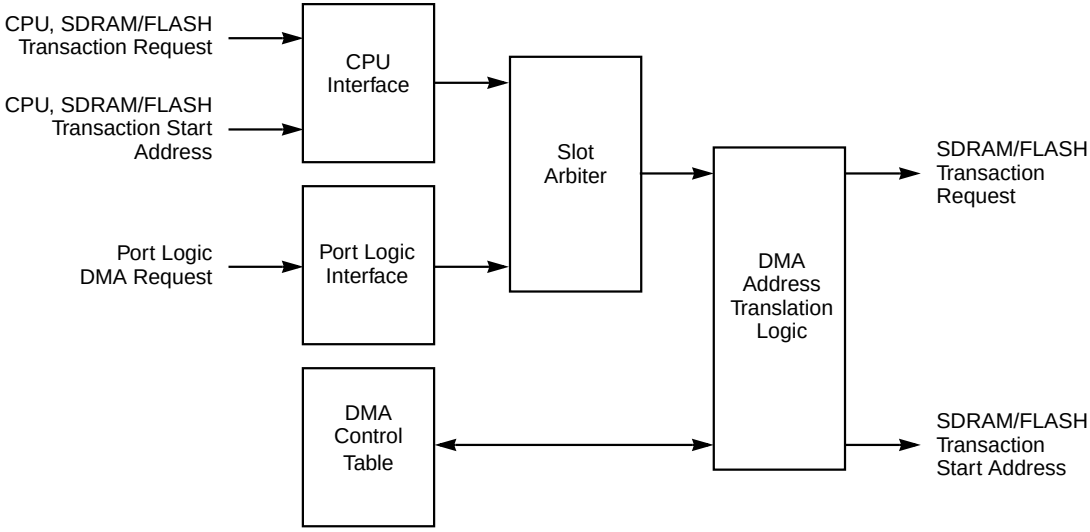
Index		Byte Address	Physical Address		
Decimal (Hex)	**Binary**		**Row (9 bits)**	**Bank Select**	**Column (8 bits)**
18 (0x12)	0b0000 0001 00 10	0x6600	0b0 0000 0010	01	0b0000 0000
1302 (0x516)	0b0101 0001 01 10	0x1e7200	ob0 1010 0010	11	0b0000 0000
2133 (0x855)	0b1000 0101 01 01	0x3f0400	0b1 0000 1010	10	0b1000 0000
3 (0x3)	0b0000 0000 00 11	0x1e00	ob0 0000 0000	01	0b1000 0000
146 (0x92)	0b0001 0100 01 10	0x36600	0b0 0001 0010	01	0b0000 0000
4090 (0xFFA)	0b1111 1111 10 10	0x5fd600	0b1 1111 1111	01	0b0000 0000

3.2 DMA Control

DMA control (Figure 3.5) maintains the DMA transfer information and does the hierarchical arbitration among port logic. DMA control includes five major blocks:

- CPU interface
- Port logic interface
- Slot arbiter
- DMA control table
- DMA address translation logic

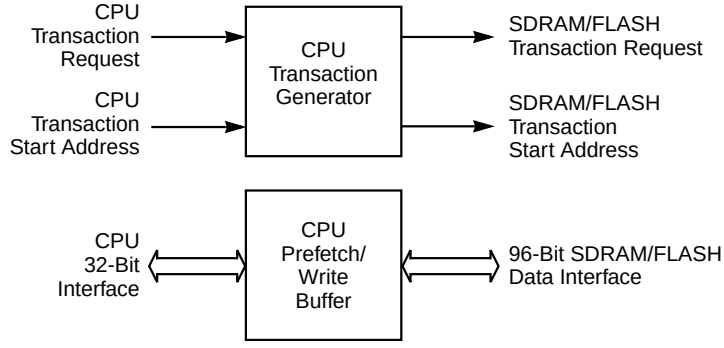
Figure 3.5 DMA Control Block Diagram



3.2.1 CPU Interface

The CPU interface (Figure 3.6) accepts CPU memory transaction requests. A CPU memory transaction can be an instruction fetch or data load/store in bytes, half words, 32-bit words, or bursts. The CPU interface includes the logic to do the bus width conversion and transaction filtering. This includes the word address and byte lane select signal generation, prefetch buffer and write buffer management, and so on.

Figure 3.6 CPU Interface Block Diagram



3.2.1.1 CPU Transaction Generator

The CPU transaction generator does the bus protocol translation between the 96 bit memory bus and the 32-bit CPU bus.

The transaction generator decides whether there is a hit in the prefetch buffer for the next instruction/data fetch. The memory system is aligned with the memory bus width (96 bytes per page). Address bit A[25] is used to fill the prefetch buffer with burst read. Address bit A[25] is set LOW for a simple byte/word/double word read only and the prefetch buffer is not altered. The CPU fills the prefetch buffer to be used later. If A[25] is used to read random data (not within the 96-byte memory boundary), it results in unnecessary memory access on the SDRAM. For maximum benefit, use A[25] to fill the prefetch buffer if consecutive data is to be accessed. All other random data should be accessed with A[25] LOW.

To increase the memory system performance, the CPU address space is organized the same way as the packet buffer space. Therefore, address translation logic is needed in the CPU interface. The CPU transaction generator is responsible for the address translation. In this way, the hardware hides the spiral or block memory structure from the CPU and port logic.

3.2.1.2 CPU Prefetch/Write Buffer

The prefetch/write buffer does the byte gathering and disassembly to bridge the 32-bit and 96-bit difference. To support nonaligned cache refill or burst write, the prefetch/write buffer size is 96 bytes long. Table3.3 lists the prefetch buffer requirements.

Table3.3 Prefetch Buffer Requirements

Corners	Requirements	
	In Prefetch/Write Buffer	**Not in Prefetch/Write Buffer**
Sequential Read (A[25] = 1)	Word output from Prefetch Buffer.	Update prefetch Buffer from SDRAM, and then output data to CPU from Prefetch Buffer.
Non-sequential Read (A[25] = 0)	Word output from Prefetch Buffer.	Word output from SDRAM directly.
Consecutive Write	Update Write Buffer (and Prefetch Buffer if it is a hit there) and then flush write buffer to SDRAM.	Update Write Buffer (and Prefetch Buffer if it is a hit there) and then flush write buffer to SDRAM.
Nonconsecutive Write	Update SDRAM directly (and Prefetch Buffer if it is a hit there).	Update SDRAM directly (and Prefetch Buffer if it is a hit there).

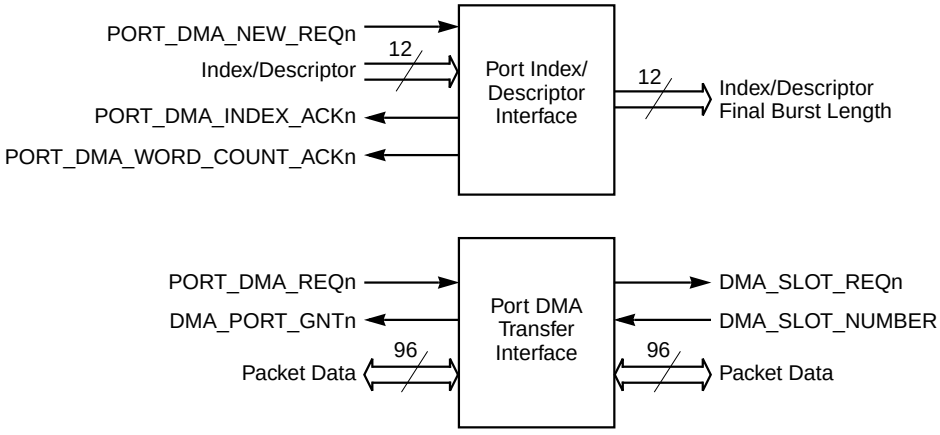
Address comparison logic decides whether there is a hit in the prefetch window. If there is a hit, the CPU reads the data from the prefetch buffer. If there is a miss for a sequential read (address bit A[25] is 1), the entire page from the memory is loaded from the SDRAM to the prefetch buffer and the CPU gets the word from the prefetch buffer. For a nonsequential read (address bit A[25] is 0), if there is a hit, the CPU gets the word from the prefetch buffer. In case of a nonsequential read miss, the SDRAM is accessed directly for the CPU read word.

The write buffer is loaded with the incoming data. If the next CPU transaction happens to be a write to the next consecutive address, that data is also stored in the write buffer. All such consecutive writes are done to the write buffer, until the next nonconsecutive write or a read operation, or the write buffer reaches the 96-byte boundary whichever happens first. The write buffer is written out to the memory in burst mode. In case of a write, if it is a hit in a prefetch buffer, the prefetch buffer is updated with the write data.

3.2.2 Port Logic Interface

Because the memory transaction queue, the DMA request, and the DMA transfer occur at different times, it is necessary to divide the port logic interface into two portions, the port index/descriptor and the DMA transfer interface. Figure 3.7 is a block diagram of the port logic interface.

Figure 3.7 Port Logic Interface Block Diagram



3.2.2.1 Port Index/Descriptor Interface

The port index/descriptor interface accepts the DMA new index request from the port logic, collects the index/descriptor, and stores the bank address for future use. Once its slot time arrives, the slot arbiter passes this information to the memory controller.

3.2.2.2 Port DMA Transfer Interface

The DMA transfer interface accepts the port slot number from the memory transaction queue and does the DMA transfer to the port. At the same time, the DMA_PORT_GNTn is asserted to identify the port doing the DMA transfer.

3.2.3 Slot Arbiter

The slot arbiter uses a hierarchical arbitration scheme. There are two levels of hierarchy and 5 priority groups to support 5 transaction types:

- Gigabit ports transmit
- Gigabit ports receive
- Megabit ports transmit
- Megabit ports receive
- CPU/Refresh

Within each group, ports are arbitrated in a round robin fashion. The transmit group has a higher priority than the receive group, and the CPU group has a dynamic priority. A priority timer is associated with the CPU. Each CPU access resets this timer and starts timing. If there is a time out from this timer, theCPU has the same priority as the megabit port transmit group. If there is no time out, the CPU has the same priority as the megabit port receive group. In this way, the CPU does not starve for the instruction. Refresh is associated with the CPU group. Any unused megabit port slot can also be allocated to the CPU. In the worst case, if the port logic requests DMA at the same time, this slot arbitration policy can still meet the performance requirement. Between the gigabit port and megabit arbitration rings, every two gigabit port services will have only one megabit port service. Transmit ports always have the highest priority. With the same type of port, every two transmit port services have one receive port service. Within each port service ring, there is a special last receive/transmit service latch. They are used as the next service pointer to separate serviced and nonserviced ports within each ring. Also, banks are used to prioritize the port service and maximize the SDRAM bandwidth.

3.2.4 DMA Control Table

The DMA control table records the relevant information of each port for the DMA transfer. The transfer bank contents of the table for each port are as follows:

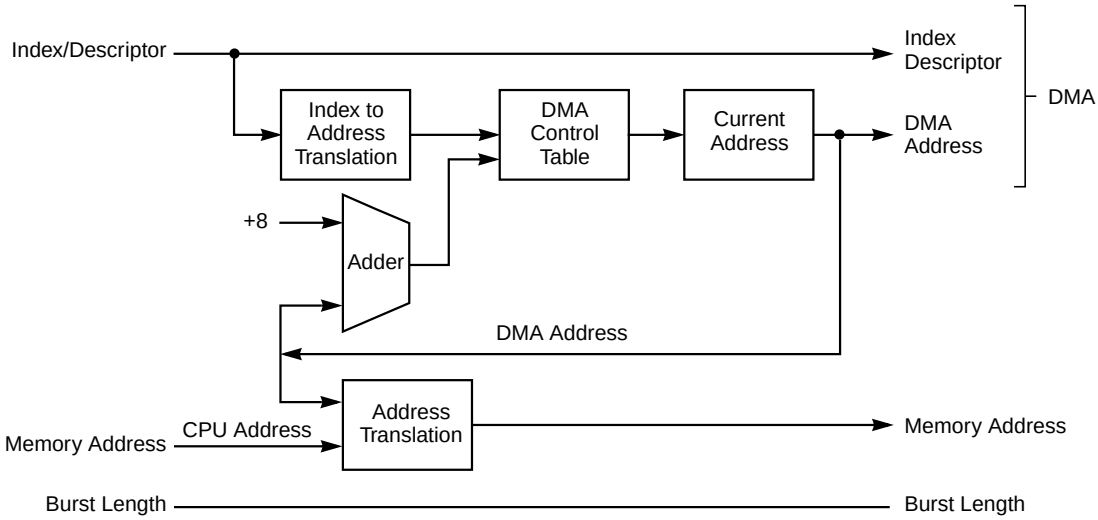
- Port ID
- DMA Transfer Byte Count
- Burst Length
- Byte Mask
- Read/Write (transfer direction)
- Next DMA Start Address

3.2.5 DMA Address Translation Logic

The DMA address translation logic is responsible for the current SDRAM address translation from index/descriptor. The address translation is directly from the index/descriptor to the SDRAM row, column, and bank

select address. The default burst length is 8 words per transfer. Figure 3.8 is a logic diagram of the DMA address generation.

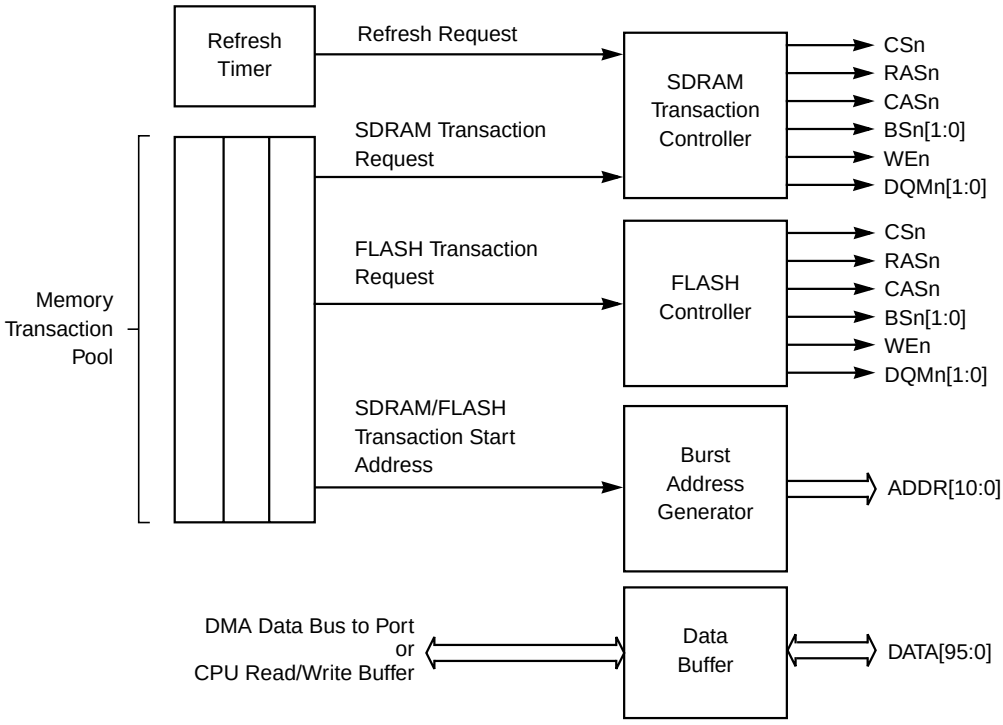
Figure 3.8 DMA Address Generation Logic Diagram



3.3 Memory Control

Figure 3.9 is a block diagram of the memory controller interface. Memory control provides control interfaces to three 64 Mbit SDRAM chips and Flash memory. The memory bus is organized in a 96-bit width. It takes advantage of the four bank organization in commercial SDRAM chips. To increase performance, packet data is evenly distributed across all internal memory banks by using a spiral scheme. In this way, the overhead cycles associated with each transfer can be evenly distributed along the burst. The longer the burst length, the better the overhead is distributed, but the size of the port FIFO is limited.

Figure 3.9 Memory Controller Interface Block Diagram



3.3.1 SDRAM Initialization

Commercial SDRAM chips have several power-up sequence requirements:

1. During power up, all input signals must be held in “NOP” state and the clock must be started at the same time.

2. After power up, a pause of at least 100 µs is required and DQM and CKE signals must be held HIGH to ensure the DQ output is in high-impedance state.

3. All banks must be precharged.

4. A minimum of eight autorefresh cycles is required to stabilize the internal circuity of the device.

5. The Mode register set command must be asserted to initialize the mode register.

During reset assertion, the memory interface performs boot configuration to the Flash configuration register by sampling external data bus bits [73:68]. Boot configuration decides the read/write wait state, and bus turnaround cycle time.

The memory interface module initializes memory after reset is deasserted as follows:

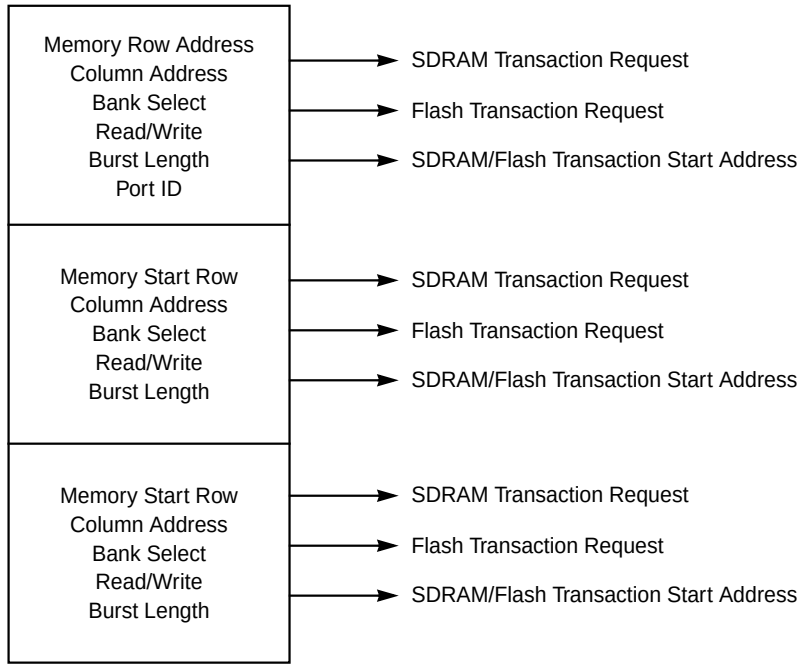
1. Precharges all the banks
2. Autorefreshes eight times
3. The Mode register set command initializes the SDRAM mode registers

The value in the SDRAM_MODE_REG register is transformed into a SDRAM command.

3.3.2 Memory Transaction Pool

To increase the memory bus performance, a three entry memory transaction pool is used as shown in Figure 3.10. Each pool entry has all the information required to issue a memory transaction, including memory start row, column address, bank select, byte select, read/write, and burst length. This pool structure allows memory control to overlap SDRAM overhead of different transactions.

Figure 3.10 Memory Transaction Pool



If a second transaction uses a different bank from the first transaction, the second transaction can be launched early to overlap the overhead of both transactions. After the first transaction's column address is issued, $[((t_{cas\ lantency}$ or $t_{wr})$ + burst length$)_{1st} - ((t_{cas\ lantency}$ or $t_{wr})$ + burst length$)_{2nd})]$ cycles later, the second transaction can be launched. If the second transaction uses the same bank as the first transaction, the second transaction waits for the first transaction's bank precharge to be finished. After the first transaction goes to precharge, the transaction can be removed from the transaction pool, and all the entries move forward one stage.

3.3.3 SDRAM Control

SDRAM control generates all SDRAM transaction timing, including read, write, burst read, burst write, precharge, mode register set, refresh, and power down. All the timing is programmable by means of the SDRAM timing register. The SDRAM control must also generate the correct command sequence to hide the SDRAM precharge overhead. All the read/write transactions precharge automatically. Because the possibility

for two succeeding transactions with the same bank select and row address is very low, there is no need to keep one bank active and expect the next access to be in the same bank and the same row. The memory system uses a 26 port time sharing scheme, therefore, the control does not support a long burst, such as full page. The default burst length is 8 words. This burst length is a compromise between the port FIFO size and the SDRAM performance characteristics. A distributed refresh scheme is used. A refresh timer issues an autorefresh request every 64 µs. The SDRAM will refresh one row at a time. Internally, the SDRAM has a refresh row address counter, and it refreshs one row for each autorefresh request.

3.3.4 Flash Controller

The Flash memory is accessed only during boot time. The Flash interface is a subset of the SDRAM interface. Due to the long access time, 70 to 120ns, accesses to the SDRAM and Flash are mutually exclusive. The Flash memory controller supports 2 Mbytes to 8 Mbytes, 8 bit width, 3.3 volt single supply devices, with embedded algorithm capability, such as AMD's AM29LV008B. The Flash memory controller can do Flash memory read, erase, and program. It also has programmable wait states, and turnaround cycle capabilities. Table 3.4 lists suggested Flash memory.

Table 3.4 Suggested Flash Memory

Manufacturer	Part Number	Description
AMD	29LV102	3.3 V only, Boot Block 2 Mbit, 256K x 8
AMD	29LV104	3.3 V only, Boot Block 4 Mbit, 512K x 8
AMD	29LV108	3.3 V only, Boot Block 8 Mbit, 1M x 8

Access of the SDRAM and the Flash are mutually exclusive. The multiplexed SDRAM/Flash signals are listed in Table 3.5

Table 3.5 Multiplexed SDRAM/FLASH signals

SDRAM Access	Flash Access
SDRAM_DATA[7:0]	FLASH_DATA[7:0]
SDRAM_DATA[27:8]	FLASH_ADDR[22:0]
SDRAM_DATA[31:28]	Last value or all ones
SDRAM_ADDR[11:0]	Last value or all ones
SDRAM_RAS	1
SDRAM_CAS	FLASH_OE
SDRAM_WE	FLASH_WE
SDRAM_CS	1
FLASH_CS	FLASH_CS
FLASH_OEN	FLASH_OEN

The maximum size of Flash memory is 8 Mbytes. When using 2 Mbyte Flash memory devices the 2 Mbyte space is duplicated 4 times. When using 4 Mbyte devices the space is duplicated 2 times. Table 3.6 lists the densities and address lines.

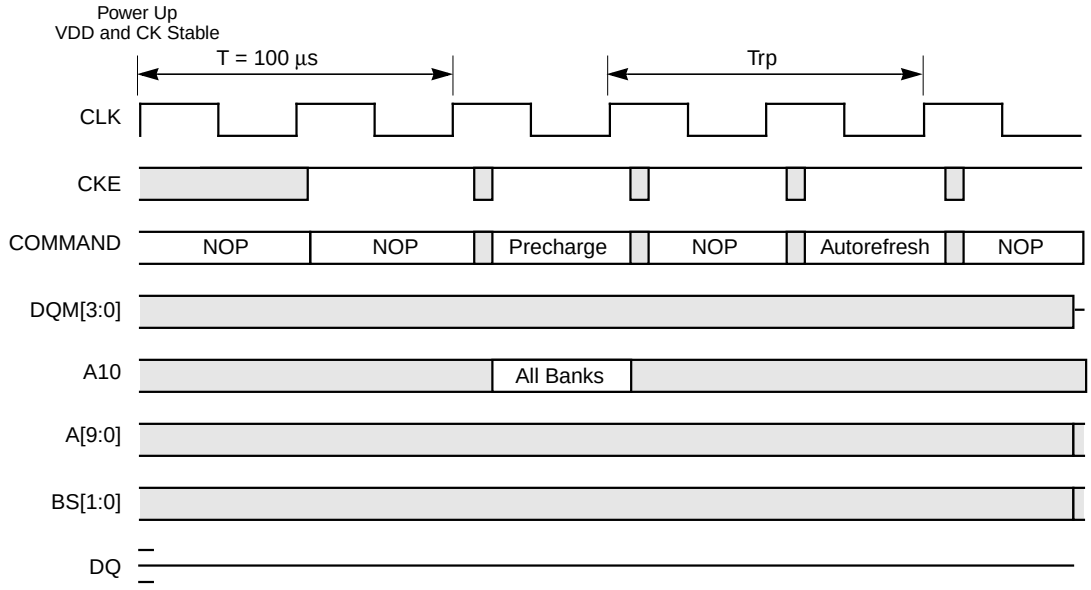
Table 3.6 Flash Density

Flash Density	Address Lines
8 Mbits (1 Mbyte)	FLASH_ADDR[19:0]
16 Mbits (2 Mbytes)	FLASH_ADDR[20:0]
32 Mbits (4 Mbytes)	FLASH_ADDR[21:0]
64 Mbits (8 Mbytes)	FLASH_ADDR[22:0]

3.3.5 Memory Transaction Timing

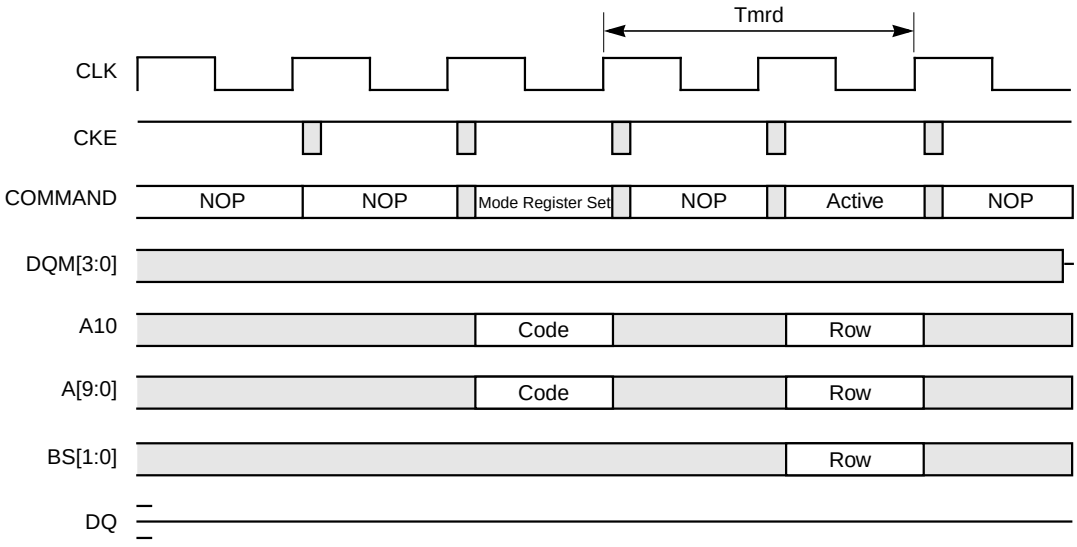
The following figures show the memory transaction timing.

Figure 3.11 SDRAM Initialization, Precharge All Command



Note: COMMAND = [CSn, RASn, CASn, WEn]

Figure 3.12 Load Mode Register Command



Note: COMMAND = CSn, RASn, CASn, WEn

Figure 3.13 4 Bank Interleave 8-Word Burst Read

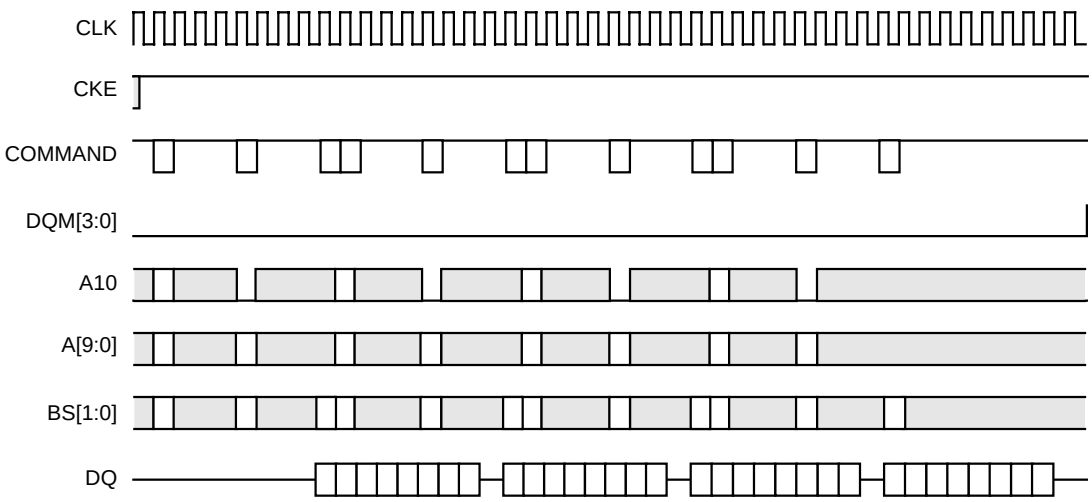


Figure 3.14 4 Bank Interleave 8-Word Burst Write

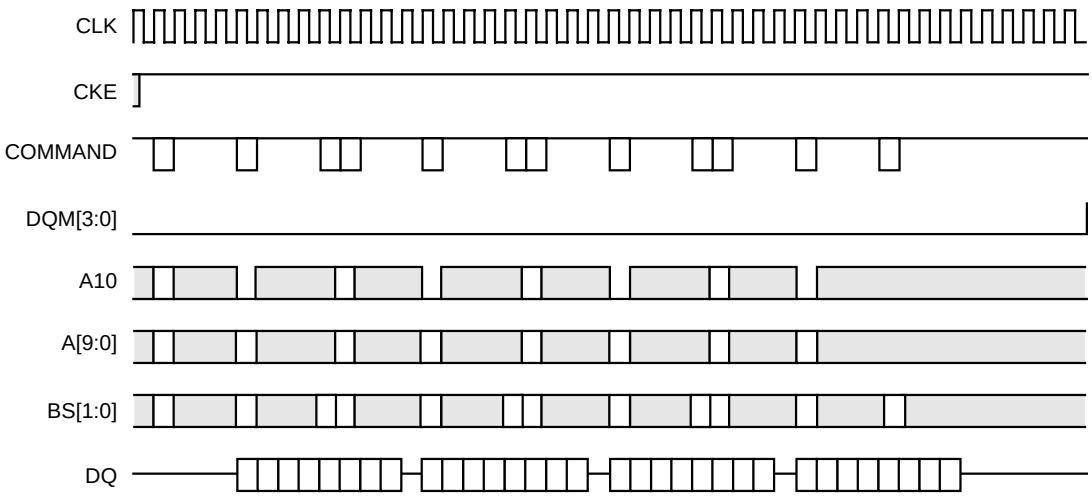


Figure 3.15 Two Bank Interleave Read

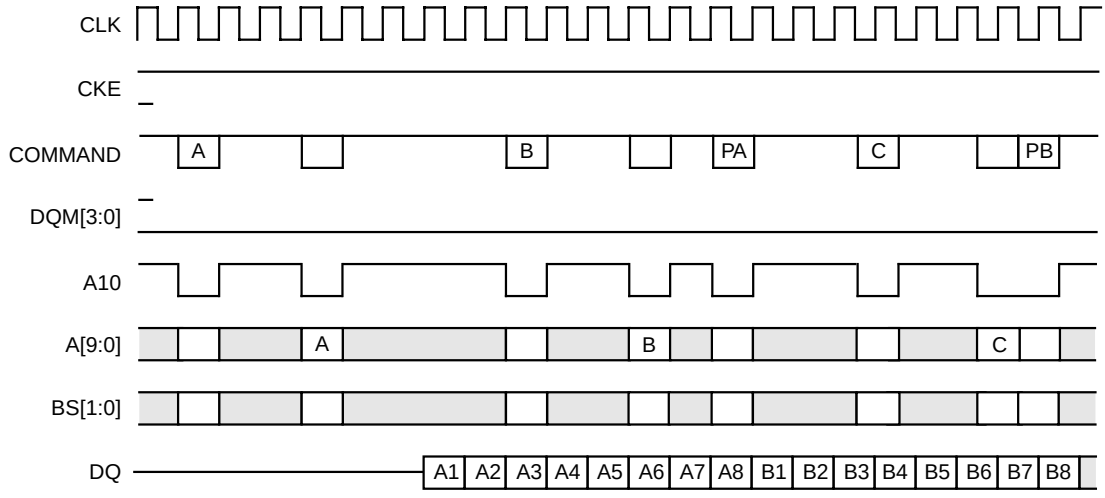


Figure 3.16 Three Bank Interleave Read

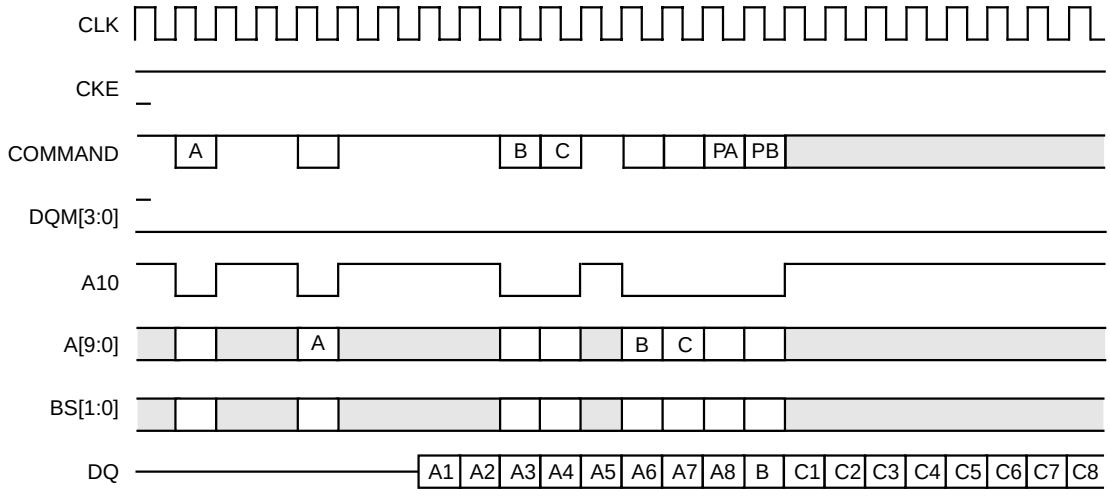
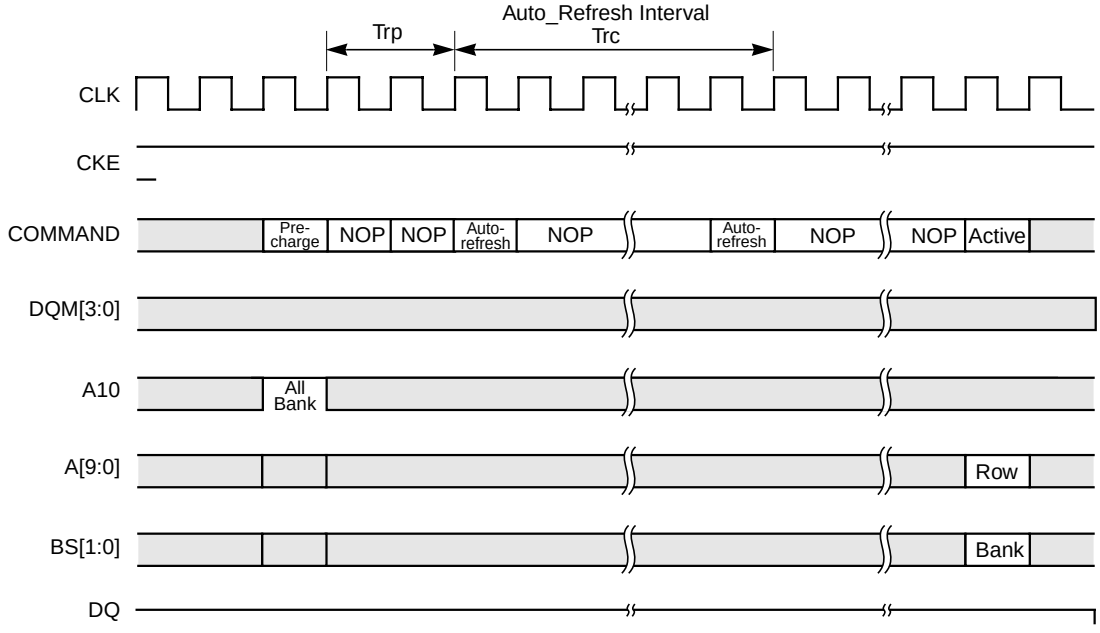


Figure 3.17 Auto_Refresh Command



Chapter 4
Buffer Management

This chapter describes buffer management of the L64324 24 x 2 Fast Ethernet Intelligent Switch and consists of the following sections:

4.1 Buffer Structure

The buffer structure is based upon a block of shared memory which is divided into fixed sized buffers. A buffer is capable of completely containing a maximum sized packet (1522 bytes). As packets arrive at the switch, they are stored in shared memory, a forwarding decision is made, and the packets are queued for transmission on one or more output ports.

Each output port has a transmit descriptor ring. A packet is queued to a given output by creating a transmit descriptor and placing it in the descriptor ring of the output port. A descriptor contains a pointer to the location in memory where a packet is stored. The ring also maintains the sequence order that is used to transmit packets. The number of packets queued to a given output port is equal to the number of consumed descriptors.

There are a total of 4096 available packet buffers. At any time the number of available buffers is reflected in the Count of Buffers Available register (CBA). As packets are received and queued for transmission, the available buffer count decreases. As packets are transmitted, the CBA increases.

4.1.1 Output Rings

Each output port has a high and low-priority transmit queue. Each output queue has a finite number of transmit descriptors. The number of descriptors in each ring can be individually controlled. The queue is arranged as a ring such that after operating on the last entry in the queue, the queue wraps around to the start. Each output ring has its own packet buffered count, for example, DUPyn. DUPy0 is the count of the number of packets queued to the default priority ring for port y. DUPy0 is also the number of descriptors consumed in priority ring 0 for port y. DUPyn = Descriptors Used in Priority ring n on port y.

When the number of allocated descriptors becomes exhausted, no additional packets can be added to the ring and the packet is discarded if it has no other destination ports. Each priority ring operates independently. The availability of descriptors in one ring has no effect on the queuing operations of the other ring for that port. It is as if a separate physical port existed for each ring. The physical existence of DUPyn is only necessary to implement output based flow control. All that is important is that the number of used and remaining descriptors be determined at each port.

4.1.2 Ring Exhaustion

When all the descriptors in a given ring have been used, no additional packets can be queued to that ring. Packets destined to the output ring are then dropped. Output flow control can be used to avoid packet loss. When the ring reaches a programmed count threshold, ports attempting to add packets to the ring are flow-controlled. Regardless of the threshold, packets continue to generate descriptor entries until all the descriptors are exhausted. The descriptor creation process is independent of the flow control mechanism. The flow control mechanism can be enabled on a per port basis.

4.2 Buffer Control

The registers and register fields listed in Table 4.1 act upon the buffer control process.

Table 4.1 Buffer Control Registers and Register Fields

Control Variables[1]	Used By	Description
Index Buffer Pointer	Port Logic, ABIT, Transmit Descriptor	This 12-bit value points to one of 4096 buffers in external memory.
CBA	ABIT, Flow Control logic	Count of the remaining buffers available in shared memory.
DUPyn	Transmit Descriptor, Flow Control logic	Count of buffers currently queued for transmission at output port y for priority queue. This is actually the number of Descriptors Used for port y priority ring n.
OFCHyn	Flow Control logic	When DUPyn ≥ OFCHyn*8, flow control is initiated at any input port that attempts to add a packet to the output queue and OFCx_ENABLE is set.
OFCLyn	Flow Control logic	Output port initiated flow control is terminated when the transmit descriptor ring count (DUPyn) drops below OFCLyn*8.
OFCx_ENABLE	Flow Control logic	This per input port enable controls whether output initiated flow control can flow control the given input port x.

1. Where: x = input port number, y = output port number, n = priority ring 0 or 1.

.

4.2.1 Unicast Forwarding

When a unicast packet arrives and there is an available buffer (CBA > 0), the packet is buffered. Otherwise, there are no memory resources available and the packet is not buffered and therefore discarded. Flow control may also have been invoked prior to this point to avoid packet loss. Once a packet is completely stored in memory, the forwarding function determines which output port, if any, the packet is destined for. If there is not a descriptor available at the destination port, the packet is discarded. Otherwise, a transmit descriptor entry is created for the

packet. If the destination descriptor ring has reached the OFCHyn threshold and OFCx_ENABLE is set, the input port initiates flow control.

When a unicast packet is transmitted, the index associated with the buffer that contains the packet is returned to the index table and CBA is incremented.

4.2.2 Multicast Forwarding

When a multicast, broadcast, or a soon to be flooded unicast packet arrives and there is an available buffer (CBA > 0) the packet is buffered. Otherwise there are no memory resources available and the packet is not buffered and therefore discarded. Flow control may have been invoked prior to this point, to avoid packet loss. Once the packet is completely stored in memory, the forwarding function determines which output port or ports, if any, the packet is destined for. Unlike the unicast packet, the multicast packet may be queued at many or all of the output ports. Once the forwarding function has generated the set of destination ports, the following is applied to each destination port simultaneously:

- If there is not a descriptor available at the destination port the packet is discarded. Otherwise, a transmit descriptor entry is created for the packet.
- If the destination descriptor ring has reached the OFCHyn threshold and OFCx_ENABLE is set, the input port initiates flow control.

When the last port transmits the multicast packet, the buffer is cleared and CBA is incremented.

4.3 Available Buffer Index Table (ABIT)

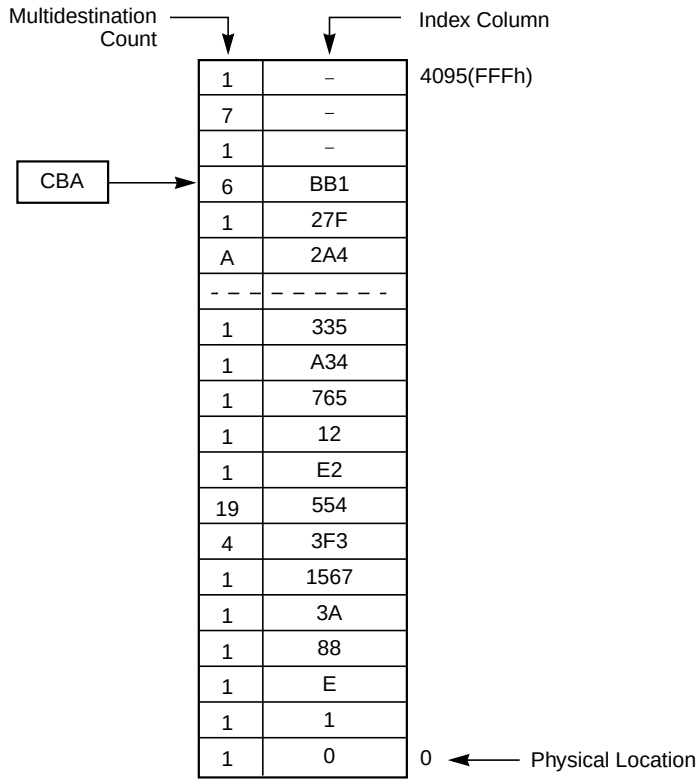
The total number of available buffers, and the specific individual buffers that are currently available to the system, are managed using the Available Buffer Index Table (ABIT). The ABIT acts as a shared depository of available packet buffers in memory. When a port requires a buffer, an index is popped from the top of the ABIT and allocated to the port. When a packet has been transmitted, the buffer is released and made available to the system by pushing the index back on top of the ABIT.

There are 4096 entries in the ABIT. Each entry in the ABIT is an index to a physical memory buffer. The physical size of each buffer is 1536 bytes in size.

Initially, the ABIT contains a single12-bit entry for every buffer in the system. The number of entries in the table at any time depends on the number of packets buffered in the system. Before the corresponding physical buffer memory location can be accessed, the index must be converted to a physical address.

The ABIT shown in Figure 4.1, also contains a column for the Multidestination count. This column is used to track the number of destinations associated with a given packet.

Figure 4.1 Available Buffer Index Table (ABIT)



Note: There is no relationship between the Multidestination count and index columns

At startup the index table is initialized to contain all 4096 buffer index entries. Because the CPU has the ability to read and write every location within the ABIT, the CPU initializes the ABIT. Once the table is initialized and ports are enabled, each port requests a buffer. For each index allocated, the Count of Buffers Available (CBA) register is decremented (the CBA pointer moves down the table). The CBA tracks the overall number of buffers that are available and triggers flow control events.

As shown in Table 4.2, the CPU should initialize the ABIT so that adjacent indexes are not positioned in the same bank initially. A sequential count of 0, 2, 4, 6, 8,….4094,1, 3, 5, 7….4095 satisfies this requirement.

Table 4.2 Abit Memory Structure Definition. Base Address = 0x4000.7FFFh

Bit	Name	R/W	Default	Description
31:21	Reserved		X	
20:16	Multidestination Count			Count of number of destinations that transmit the associated packet
15:12	Reserved			
11:0	Index			Pointer to one of the 4096 1536-byte buffers in packet memory

Note that the Multidestination count and the index have no relationship.

4.3.1 ABIT Logic

The ABIT logic is responsible for both the allocation and deallocation of indexes in response to port requests. When an index is allocated to a port, the CBA is decremented and the location in the ABIT that contained the index is not altered. Only a copy of the entry is sent to the port. When an index is returned to the ABIT, the CBA is incremented and the Index is written to the entry pointed to by the CBA. Only when an Index is written to the table are the previous table entry contents altered. Note that since the ABIT also contains the multidestination count, it is necessary to perform a read-modify-write, or to enhance the array so that there exists the ability to control writes to the two columns individually. This would be similar to a byte write function but it would operate on a column width basis.

Because index entries are deallocated back to the ABIT at random times by each of the individual ports, any detectable sequence that may have initially existed in the ABIT quickly becomes randomized.

Once packets begin arriving, additional indexes must be supplied to a given port at a rate that is faster than it takes to receive back to back minimum sized packets. As indicated in Table 4.3, it is the time between minimum sized packets that is the most demanding.

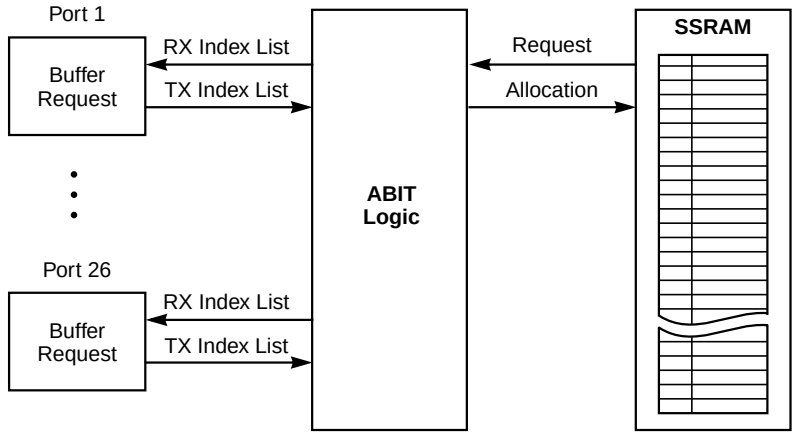
Table 4.3 Time Between Minimum sized packets

Port Speed	Time Between Min. Sized Packets
10	67.2 µs
100	6.72 µs
1000	672 ns

At a system level, the ABIT logic must be capable of allocating one Index every 152 ns in order to maintain media speed. Also, the 100 Mbit and Gigabit ports must be serviced at a rate of 6.72 µs and 672 ns, respectively. For example, a new index could be required every 672 ns at each Gigabit port. This is easily achieved with relatively slow clocked logic as described later. When a packet is filtered or successfully transmitted, the buffer that was used to contain the packet must be deallocated. Buffer deallocation is simply the process of returning the buffer index back to the ABIT. Note that the above allocation rates must still be satisfied in the presence of the deallocation and multidestination processes.

There are many ways in which to implement the ABIT and its allocation/deallocation functions. The table itself is implemented in a separate SSRAM. With the SSRAM approach, a small register cache is added to reduce accesses to memory. Figure 4.2 is an example of the implementation.

Figure 4.2 ABIT Logic



4.3.2 ABIT Arbiter

The performance demands of the ABIT arbiter are not great, as a new index request need only be granted every 152 ns. However, there are also accesses that decrement the multidestination bit. The ABIT arbiter is implemented as a rapid response arbiter. Since there is more than enough bandwidth available from the arbiter and the ABIT, all that is important is that latency concerns are addressed. For example, Gigabit ports can request an index ten times more often than the Megabit ports and should therefore be given priority. Also, the CPU should be given priority since it essentially enters a spin loop while waiting for an index. The resulting arbiter looks just like the round robin arbiter with the exception that the CPU is given the highest priority over the Gigabit port requests.

4.3.3 Receive Port Index Use

After packet reception, the index used for a given packet must be retained until it is determined that the packet will be forwarded to at least one destination. Should the packet contain errors or if the packet is not destined for the CPU or if the packet is to be filtered, then the buffer used to contain the packet must be released back to the available pool of buffers or reused for the next packet. Once the packet is received as good and it is determined that the packet will be forwarded to at least one destination, the receive port logic no longer needs to retain the

index. Once the logic has made this determination, the source port is signaled that it may discard or reuse the index. The transmit descriptor logic will store the index in a descriptor. At the completion of a successful packet transmission, the transmit port(s) will deallocate the buffer by returning the index contained in the descriptor back to the ABIT. If an error packet is received on a port that is being reflected to a mirror port, then the packet is forwarded to the mirror port.

4.3.4 CPU Interface to the ABIT

The ABIT appears as a 32-bit wide memory structure containing 4096 entries. Each 32-bit entry has the format shown in Table 4.4. This structure only supports full 32-bit accesses by the CPU. Byte accesses are not supported.

Table 4.4 ABIT Entry Format, Address = 0x3426.4000–0x3426.7FFF

31:21	20:16	15:12	11:0
Reserved	Multidestination Count	Reserved	Address Index

The CPU is responsible for initializing the ABIT such that 4096 unique indexes are loaded into the ABIT. The multidestination count field should be set to 1 for each entry. Once Initialized, the ABIT logic maintains and operates upon the entries.

The CPU also interfaces to the ABIT to request and return Indexes during normal operation.

4.4 Flow Control

Flow control is invoked under two conditions:

1. When the overall shared buffer availability drops below a programmed threshold.
2. When a given output port has queued a programmed number of buffers, any attempt to add buffers causes the input (source) port to be flow controlled.

Both flow control types are OR'ed, such that any one type can result in flow control independent of the other.

4.4.1 System Buffer Depletion Flow Control

System Buffer depletion flow control is performed on a system wide basis when the number of remaining buffers (CBA) drops below the programmed value contained in the System Flow Control Low Threshold field (SFCL) of the System Flow Control Register.

Ports can operate in one of three states relative to flow control:

1. IEEE 802.3x flow control enabled – pause packets are used to xoff traffic.
2. Back pressure – jamming is used to cause collisions and xoff traffic.
3. Neither IEEE 802.3x or flow control supported – there is no link or physical layer means to limit traffic.

When the number of overall remaining buffers in the system falls below SFCL as shown in Figure 4.3 and Table 4.5, back pressure or flow control is initiated on corresponding ports. Back pressure or flow control continues until the overall remaining buffers increase to the point that the System Flow Control High threshold (SFCH) is reached or exceeded. Once SFCH is reached, system depletion flow control is terminated on all ports. However, if ports continue to satisfy conditions for output depletion flow control, flow control continues on those ports.

Figure 4.3 System Buffer Depletion Flow Control

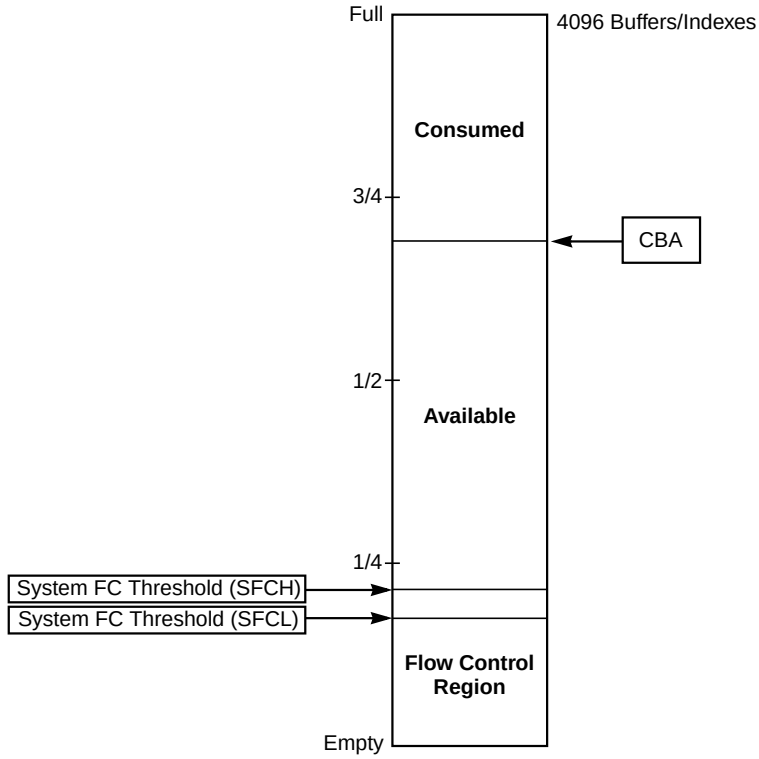


Table 4.5 System Flow Control

Event	Condition
System Flow Control Initiated	CBA ≤ SFCL
System Flow Control Disabled	CBA ≥ SFCH

4.4.2 CPU and Depletion Flow Control

Before the CPU can send a packet, an index must first be allocated to the CPU. Then the packet must be created in the corresponding buffer in memory. The CPU issues a request to the ABIT for an index. The ABIT continues to service requests from ports and the CPU independently of the depletion flow control threshold. As long as there are buffers available, Indexes are allocated to the CPU and to ports. Packets should cease arriving on those ports that have initiated flow control or back

pressure. As a result, indexes are no longer requested. The CPU does not discontinue requests as there is no concept of flow control relative to the CPU, nor is there a reason to create such a mechanism to hold off the CPU. Only when the system has exhausted its buffers should the CPU not be serviced.

4.4.3 Output Queue Flow Control

It is often necessary to limit the number of packets that are permitted to be queued to a given output. For example, given the three ports (A, B, and C), with port B having much less bandwidth than ports A and C, port B will not be able to transmit packets as quickly as port A is queuing them to be sent. Therefore, Port B will fall behind and its transmit queue will become increasing long. Without some sort of limit, port B will consume all of the available buffers. Given that the L64324 supports 10 Mbits/s, 100 Mbits/s, and 1000 Mbits/s ports, this is especially important. Limiting the number of buffers that can be queued to an output also limits the latency through the switch. The deeper the transmit queue, the greater the latency.

The size of an output queue's transmit descriptor ring inherently limits the queue. When all the descriptors in the ring are exhausted, packets destined for the output are discarded. Ideally, when flow control is enabled system wide, it is undesirable to discard packets. To avoid discarding packets, output-initiated flow control is used to Xoff the traffic to the output queue.

Each output queue has a set of low and high output flow control thresholds. Since there are two queues per output, one for high priority and one for default priority, there are a total of four thresholds. All four threshold fields are contained in a single register. When the number of buffers (CBOyn) and the number of used transmitted descriptors used for priority ring of port y equals or exceeds the Output Flow Control High Threshold (OFCHyn) for the given port, any source port that attempts to forward a packet to the queue will be flow controlled. As long as CBOyn equals or exceeds OFCHyn, any new port which attempts to add a packet to the queue will also be flow controlled. Packets will continue to be added to the queue as long as there are descriptors available. When the descriptors are exhausted, additional packets are discarded.

When CBOyn falls below OFCHyn, new ports that attempt to add to the output queue, are not flow controlled. However, ports that are currently

flow controlled continue to be flow controlled until the number of buffers in the output queue falls below the value in the Flow Control Output Low Threshold register (OFCLyn).

Setting the OFCHyn threshold greater than the number of descriptors allotted to the given port and priority ring disables this type of flow control. Uplink ports or ports connected to other switches should be flow controlled only when overall system buffer resources are exhausted, rather than with output flow control.

It may not be desirable to initiate output flow control as a result of selected input ports. For this reason, output initiated flow control may be enabled on a per input port basis by means of the Output Flow Control Enable register. When CBOyn ≥ OFCHyn, and a given input (source) port attempts to add a packet to the descriptor ring, that input port is only flow controlled if its enable bit is set in the Output Flow Control Enable register.

It is possible for several output queues to exceed their thresholds such that when a given input port attempts to add a packet to the queue of each output, the output ports all initiate flow control on the same input(s). When a given output port falls below the Output Flow Control Low threshold (OFCLyn), flow control should be disabled on the input ports that were exclusively flow controlled by the given output port.

Each output queue should contain a 26-bit latch. Each of the latch's 26 output bits corresponds to one of the 26 possible ports. When an input port attempts to add a packet to an output port that has exceeded its threshold (CBOyn ≥ OFCHyn), the corresponding port bit of the latch becomes set. Each latch bit is wire OR'ed with all the other like register bits of the other queues. The wire OR'ed output is then sent to the input port. As a result, several output ports can simultaneously flow control a given input port. Flow control continues on a given port until all sources of output-initiated flow control have deasserted the common signal. Output-initiated flow control should only be enabled on input ports that have a single end station attached. Output flow control should not be enabled on input ports that have shared resources, such as printers and servers, otherwise a head-of-line block results.

4.4.4 CPU and Output Flow Control

Just as the CPU can receive packets, the CPU can also send packets. When the CPU wishes to send a packet, a transmit descriptor is created. When a transmit descriptor is pending in the CPU's transmit descriptor queue, the switch logic services the CPU similarly to the way in which port receive requests are serviced. Once serviced, the switch logic uses the descriptor to ultimately generate a forwarding mask. The mask is presented to the transmit descriptor logic, which then creates descriptors for the various destination ports. If the descriptor ring of a destination port has exceeded its descriptor ring output flow control threshold (OFCHyn), the source port is normally flow controlled, if flow control is enabled on the input port. The flow control process occurs independently of the transmit descriptor generation process. Descriptors continue to be created as long as there are descriptors available. If flow control is effective, the given input port ceases sending packets to the output port and the ring eventually shrinks in size. When the number of used descriptors falls below the output flow control low threshold, flow control is terminated at the input port. If the input port does not support flow control, and packets continue to be sent to the congested output port, the descriptors eventually become exhausted and additional packets are discarded at that port.

CPU sourced packets are treated exactly the same way as packets from ports, except that the CPU is not flow controlled. Packets from the CPU continue to be added to a given output queue as long as there are descriptors available. When the descriptors in a given ring are exhausted, packets from any source are discarded.

4.4.5 Head-of-Line Blocking

Flow control can introduce head-of-line blocking when there is more than one station attached per port. For example, if there are two nodes on a given port, each node sending traffic to a different output port, when flow control is initiated for the first node, the second node is also flow controlled. For this reason flow control should generally be enabled on ports with a single end node.

4.4.6 Back Pressure

Ports operating in half-duplex mode have the option to enable back pressure. Back pressure is enabled on a per port basis by means of the Back Pressure field of the MAC configuration register. Back pressure ports are controlled exactly the same way as flow control ports and are controlled at the port.

4.5 Flow Control Monitoring

Monitoring the contents of CBA and the depth of the transmit descriptor rings, allow software to monitor the overall state of the switch from a flow control perspective. The number of pause frames received and transmitted per port is available in the counter status vector sources.

4.6 Flow Control Register Summary

Table 4.6 is a summary of the flow control registers.

Table 4.6 Flow Control Register Summary

Field or Register[1]	Used By	Description
CBA	ABIT, Flow Control logic	Count of the remaining Buffers Available in shared memory.
CBOyn	Transmit Descriptor, Flow Control logic	Count of buffers currently queued for transmission at output port y for priority queue n
OFCHyn	Flow Control logic	When $CBOy_n \geq$ OFCHsn*8, flow control is initiated at any input port that attempts to add a packet to the output queue and OFCx_enable is set
OFCLyn	Flow Control logic	Output port initiated flow control is terminated when the transmit descriptor ring count ($CBOy_{1)}$ drops below OFCLyn*8
OFCx_ENABLE	Flow Control logic	This per input port enable controls whether output initiated flow control can flow control the given input port x.

1. where, x=input port number, y = output port number, n = priority ring 0 or 1.

4.7 Typical Flow Control Configuration

The following example configuration assumes a system with one Gigabit uplink port, 12 active 100 Megabit ports, and 12 active 10 Megabit ports.

4.7.1 System Depletion Flow Control

With system depletion flow control a global threshold must be selected (SFCL) which initiates flow control. The goal in setting this threshold is to set it as low as possible without hitting the condition were the system buffers become exhausted before traffic can be flow controlled or Xoff'ed. Assuming that all ports in this example support flow control, an initial setting of 3*n or 3*25 = 75 is reasonable. Where three is the number of packets that could be received from the time flow control is signaled until the time that traffic ceases. The system high threshold should be some number greater than the SFCL value such that a large percentage of Pause packets are not transmitted relative to the bandwidth of the links assuming the switch would oscillate between Xon and Xoff. An initial value of 150 is reasonable.

4.7.2 Output Flow Control

To prevent slower speed ports from naturally accumulating an excessive number of buffers due to the potential speed mismatch with other ports, the output queues are limited. The limiting of queue depth also benefits the uplink port that is faced with an internal traffic rate that at times exceeds the output link rate. Since the output queues are limited to a given number of descriptors, a limit is reached and packets may be discarded. To prevent discarding packets, it would be preferable to flow control the source of the traffic. This is the purpose behind the output flow control thresholds. These thresholds are set on an individual output queue basis

The 1000 Megabit port in this example is the uplink port for 24 other ports and thus requires a large number of descriptors to handle 24 simultaneous traffic streams. Packets received on the uplink are

forwarded to destination ports. However, it is not desirable to flow control the uplink port when a single destination output queue becomes exhausted. Therefore, the OFCx_ENABLE bit for the uplink port is cleared. The 10 Megabit ports require a large number of buffers to handle the 100x rate adaption. This is especially important since flow control of the uplink port is not recommended..

4.7.3 Output Flow Control High (OFCH)

When a broadcast, multicast, or flooded unicast is processed, large numbers of descriptors can be consumed. At any time there can be a greater number of descriptors consumed than buffers consumed. As a result, the L64324 has 70% more descriptors than buffers. Output flow control is initiated when a given descriptor ring is nearly exhausted. As a result, the selected threshold is not necessarily related to the number of buffers in the system. It is necessary to set the Output Flow Control High threshold (OFCH) field such that with several ports sending to another port, flow control can be initiated on the source ports so packets received in the interim are not lost.

Assume that 50% of the ports can send at any time to an exhausted ring for both 10 Megabit and 100 Megabit ports, and 100% of the ports can send to the 1000 Megabit ports, and that up to 3 packets can be received before flow control can Xoff the traffic. 3 buffers*12 ports = 36 buffers, and 3 buffers*24 ports = 72 buffers. 66% of this traffic goes to the default priority ring and 33% to the low priority ring.

These values are then used in combination with the descriptor ring sizes to find the OFCHyn thresholds.

For 10 Megabit, $OFCHy_0 = 172 - 0.66*36 = 148 \div 8 = 0x12$

For 10 Megabit, $OFCHy_1 = 86 - 0.33*36 = 74 \div 8 = 9$

For 100 Megabit, $OFCHy_0 = 86 - 0.66*36 = 62 = 8$

For 100 Megabit, $OFCHy_1 = 43 - 0.33*36 = 31 = 4$

For 1000 Megabit, $OFCHy_0 = 1554 - 0.66*72 = 1506 \div 8 = 0xBC$

For 1000 Megabit, $OFCHy_1 = 777 - 0.33*72 = 753 \div 8 = 0x5E$

The low threshold is just OFCH – 1.

Chapter 5
Address Resolution Logic

This chapter provides a description of the L64324 24 x 2 Fast Ethernet Intelligent Switch Address Resolution Logic (ARL) and contains the following sections:

5.1 Overview

The Address Resolution Logic (ARL) is the heart of the switch. The ARL is responsible for the creation, maintenance, and operation of an address table. The ARL is fully autonomous and does not specifically require CPU intervention. However, the CPU can read and write to all fields of the address table and is responsible for the addition and maintenance of multicast entries.

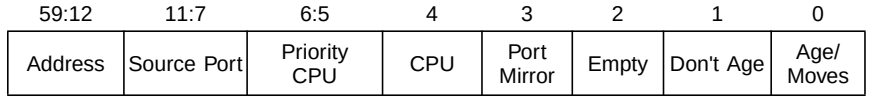
5.2 Address Table

The address table contains 4096, 60-bit word entries. The table contains two types of entries: Unicast and Multicast.

5.3 Unicast Entries

Each Unicast address entry is in one location in the table and has the format shown in Figure 5.1.

Figure 5.1 Unicast Address Table Entry



5.3.1 Age/Moves [0]

The Age/Moves bit is cleared when an address is first learned and seen as a source address. The aging logic interval sets the Age/Moves bit. When the aging routine detects that the Age/Moves bit is set, the address table is cleared and the Empty bit is set. An entry is not aged out when:

- The Don't Age bit is set
- The Port Lock bit is set for the source port in the Source Port Lock register
- Port Aging Control Register disables ageing

For each of these conditions, the Don’t Age bit is considered set and the Age/Moves bit takes on new meaning. When Aging has been disabled, the Age/Moves bit determines if moves will be allowed as follows:

	Age/Moves = 0	**Age/Moves = 1**
Don’t Age = 0	Moves Allowed	Moves Allowed
Don’t Age = 1	Moves Not Allowed	Moves Allowed

Moves are allowed for Source Port Locked ports only if the moves are to other Source Port Locked ports.

5.3.2 Don’t Age [1]

When the Don’t Age bit is set, the address is never aged. When the source port locking feature has been enabled or aging has been disabled for a given port, the Don't Age bit is considered set for all corresponding address table entries, regardless of its actual value.

5.3.3 Empty [2]

When the Empty bit is set, and the address is not a multicast address, the address table entry is free to accept a new entry.

5.3.4 Port Mirror [3]

If the Port Mirror bit is set, the packet is also sent to the port specified in the mirror port field of the Port Mirror Mask Register, in addition to the intended destination. Only the CPU sets this bit. By default this bit is cleared. This function sends all packets that contain the associated source or destination address to the mirror port, regardless of whether the packet is filtered or forwarded.

5.3.5 CPU [4]

If the CPU bit is set, the packet is sent to one of the CPU's four priority queues in addition to the intended destination. Only the CPU sets this bit. By default, this bit is cleared. Clearing the Source Port field bits also causes the packet to be forwarded to the CPU. The CPU queue is selected by means of the $Priority_{CPU}$ bits.

5.3.6 $Priority_{CPU}$ [6:5]

If the packet is to be copied to the CPU, as determined by the CPU bit or a source port of 0, then this two-bit field determines which of the four CPU receive queues the packet uses as shown below:

Priority $_{CPU}$ [6:5]	CPU Receive Queue
00	Lowest
01	Low
10	High
11	Highest

5.3.7 Source Port [11:7]

This five-bit field specifies the port on which the address was last sourced. Valid entries are 0 to 26, where port 0 is the CPU. If the field is set to 0 the packet is sent to the CPU and placed in one of the four queues as determined by the Priority$_{CPU}$ field.

5.3.8 Address [59:12]

This is the 48-bit unicast address. Bit 52 is always 0 for unicast entries

5.4 Multicast Entries

Multicast packets can be forwarded to any port, any combination of ports, or all ports. To provide this flexibility, each multicast address table entry contains a 26-bit port mask. When the port mask bit for a given port is set, the packet containing the multicast destination address is forwarded to that port as well as any other port which also has the port mask bit set. To accommodate the extra 26 bits associated with the multicast port mask, each multicast entry consumes two table entries.

5.4.1 Multicast Address Entry (First Word)

The first word of a multicast entry is shown in Figure 5.2.

Figure 5.2 Multicast Address Entry (First Word)

There are essentially two types of Multicast entries; IP Multicast and Non-IP Multicast addresses. For multicast entries other than IP Multicast, a single multicast address table entry is necessary. This one entry applies across all VLANs and is used in conjunction with the VLAN table to determine the forwarding mask.

For IP Multicast, It is necessary to specify a separate entry for each VLAN that accepts the multicast address. A given port can be a member of several VLANs. A router sends a copy of an IP multicast packet to each VLAN that has a listener. To prevent a port from receiving multiple copies of the same packet, a given port has its port mask bit set in only one IP Multicast/VLAN entry. For example, assume there are five entries all with the same multicast address but with different VLAN IDs. A given IP Multicast address table port mask bit should only be set in one of the five entries. Setting more than one bit causes duplicate packets to be delivered.

Because a multicast address can never be a source address, they cannot be learned like unicast entries. The CPU adds and removes multicast entries. Automatic hardware aging is not performed on multicast entries. As a result, there is no Empty, Don't Age, or Age/Moves bit. Examining the group bit of the address field distinguishes multicast from unicast entries. The group bit is always 1 for multicast addresses. Hence, an entry is multicast if bit 52 of the entry is set. Clearing this bit converts the entry into the unicast format. A multicast entry always has a corresponding second entry. The second entry is also converted.

5.4.1.1 VLAN_ID [11:0]

VLAN_ID is a 12-bit field containing the VLAN_ID associated with the IP Multicast MAC address. Specifically, this field supports IP multicast traffic. This arrangement allows any one of the possible VLANs to have a unique port mask for a given multicast address. For non-IP multicast addresses, this field is a don’t care.

The second word of a multicast entry contains the multicast port mask. As shown in Figure 5.3 Multicast Address Table Entry (Second Word), there is a mask position for each of the 26 ports. The second word of a multicast entry always follows the first word in the table. The format in Figure 5.3 only applies when there exists a previous multicast first word entry. If this entry does not exist, as indicated by the group bit, the second word takes the format of a unicast address.

5.4.1.2 Address [59:12]

This is the 48-bit Multicast address. The address is stored as it is written and not as it is transmitted.

5.4.2 Multicast Address Entry (Second Word)

The second word of a multicast entry is shown in Figure 5.3

Figure 5.3 Multicast Address Entry (Second Word)

59:47	46:21	20:13	12	11:7	6:5	4	3	2	1	0
Unused	Port Mask Port 26–Port 1	Unused	Priority Boost	Multicast Type	Priority CPU	EMP	Port Mirror	X	X	X

To support the two-word format of multicast entries, the 4096 entry address table contains 2048 Hash sites. A Hash site contains two unicast entries or one multicast entry.

Hashing to the table and reading two entries at a time, the multicast and unicast entries can co-exist. For example, there is no chance of hashing to the second word of a multicast entry. Hence there is no need to specifically mark the second word of a multicast entry as being such. It is enough to know that if an entry is multicast, it has a corresponding second word.

5.4.2.1 Port Mirror [3]

If the Port Mirror bit is set, the packet is also sent to the port specified in the mirror port field of the Port Mirror Mask Register, in addition to the intended destination. This function will send all packets that contain the multicast address to the mirror port, regardless of whether the packet would have been filtered or forwarded.

5.4.2.2 EMP (Enable Multicast Processing) [4]

When a known multicast packet is received from a port that has an STP state of blocking or learning, and the EMP bit is set in the address table for the multicast entry, the packet is forwarded as if the port were in the forwarding state, except that learning is not performed in the blocking state.

Specifically, if a multicast packet is received from a blocking or learning port, and the EMP bit is set in the address table, the packet is forwarded according to the multicast type and port mask. If the EMP bit is set for the multicast address and the source address entry has the CPU bit set, the packet is forwarded to the CPU.

If the EMP bit is cleared, or the multicast address is not known, the packet is not sent to the CPU or ports, regardless of the multicast type field or port mask.

The EMP bit only applies to multicast packets. When a port is in blocking or learning, all unicast and unknown multicast packets are dropped, unless the port is a source to a mirror port, in which case, the packet is sent only to the mirror port. When in blocking mode, the source address is never learned, regardless of the state of the EMP bit.

This function allows the CPU to receive any specified multicast packet on ports that are in blocking or learning STP states.

5.4.2.3 $Priority_{CPU}$ [6:5]

If the packet is to be copied to the CPU, as determined by the CPU bit or a source port of 0, then this two-bit field determines which of the four CPU receive queues the packet will use:

$Priority_{CPU}$ [6:5]	CPU Receive Queue
00	Lowest
01	Low
10	High
11	Highest

5.4.2.4 Multicast Type [11:7]

The Multicast type field has the following encoding:

[11:7]	Multicast Type	Description
00000	No copy	The packet should not be copied to the CPU
00001	Generic Multicast	Copy to CPU
00010	IGMP	IGMP packet to be copied to the CPU
00011	GVRP/GMRP	VLAN Protocol Packet to be sent to the CPU
00100	Unknown Multicast	Unknown Multicast, set by ARL – copied to CPU
00101	OSPF	OSPF packet detected by port logic, set by ARL – copied to the CPU
00110	PIM	PIM packet detected by port logic, set by ARL, copied to the CPU
00111 through 11111	Other	Defined by CPU – copied to CPU

Unknown multicast type and generic multicast type are never actually set in the address table. The Address Resolution Logic sets this entry.

5.4.2.5 $Priority_{BOOST}$ [12]

The $Priority_{BOOST}$ bit determines if the entry corresponds to a multicast group that is to be forwarded at high priority. This only has the effect of placing the packet on the high priority output queue(s). It has no effect on the priority tag field contained in the packet or the default port priority associated with the receiving port. However, this priority bit, when set, overrides the tagged field and the Port Priority Default (PPD) with respect to determining which queue the packet is placed on at each output. When this bit is cleared, it has no effect.

5.4.2.6 Port Mask [46:21]

There is a bit in the port mask field for each of the 26 ports. Setting a bit in one of the port mask field positions forwards the packet to the

corresponding port, assuming all other conditions are satisfied, such as VLAN membership, buffer consumption, and so on.

5.5 Static Entries

A static address entry is an address that is not aged and is associated with a single constant source port. Static entries are not allowed to move. Moves occur when a given address is received from a port other than the one specified in the Source Port field of the address table. Allowing moves convert static entries to pseudo static entries. Moves are controlled in this situation by means of the Age/Moves bit of the address table. The Age/Moves bit controls move operations when any of the following conditions are true:

- The Don't Age bit is set in the address table.
- The Port Lock bit is set for the source port in the Source Port Lock register.
- Aging has been disabled for the given port by means of the Port Aging Control register.

Each of these conditions also disables aging, either for a given address or for all addresses associated with a given port.

The Age/Moves bit determines if the Source Port field of the address table is updated to reflect a new source port. When a move is detected and the Age/Moves bit is set, the address table is updated with a new source port. When Aging is disabled for an entry or port, and the Moves bit is cleared, the entry is considered to be Static.

When a packet is received that contains a static entry source address, but is received on a port that does not match the Source Port field in the address table, the address table is not updated, but the packet is still forwarded as if it were received on the correct source port. So the address table still indicates that the source address is on one port, but the traffic from that source address is coming in from another port. Regardless, the traffic from the source is forwarded (unless there are VLAN differences). However, traffic destined to the source address continues to be forwarded according to the static Source Port field in the address table. Therefore, if the static end node has moved to a new port,

that end node will not receive traffic destined to it. Table 5.1 shows the conditions that determine a static entry.

Table 5.1 Static Entries

Port Don't Age bit	Don't Age Bit	Age/Moves bit	Static Entry
0	0	0	No
0	0	1	No
X	1	0	Yes
X	1	1	No
1	X	0	Yes
1	X	1	No

5.6 Source Port Locking (SPL)

The source port locking feature locks addresses to given ports to provide added security. Each unicast address in the address table has an associated source port. When a packet is received from a locked port, the packet may be filtered if its address table source port field does not match. The Source Port Locking feature protects against unauthorized moves and adds of end nodes.

Packets received on ports that are in blocking or learning mode can still result in Source Port Locking violations. Moves are also allowed just as defined for ports in the forwarding state; however, moves are only allowed if the port is in learning or forwarding state. Moves are not allowed to ports that are in the blocking state.

5.6.1 Moves

Security can be compromised when Port-Based VLANs are implemented and a given end node moves from one switch port to another. The original port may have had a PVID corresponding to a low or default security level, while the second port could have a PVID corresponding to a high priority VLAN. This mechanism prevents users from moving from one port to another.

Moves are allowed if the Age/Moves bit in the Address table is set, and the move is to another source port locked port. For this condition, a source port locking violation has not occurred. Table 5.2 lists move actions.

Table 5.2 Moves Action Table

Receiving Port	Source Address	SPL violation	Move Allowed[1]	Forwarded
Not Locked	Not Locked or unknown	No	Yes (Moved or learned)	Yes
Not Locked	Source Port Locked	Yes	No	No[2]
Source Port Locked	Not Locked or unknown	Yes	No	No[2]
Source Port Locked	Source Port Locked	Moves = 1, Yes[3] Moves = 0, Yes	Moves = 1, Yes Moves = 0, No	Moves = 1, Yes Moves = 0, No

1. Moves are only allowed if the new source port is in forwarding or Learning mode.
2. May still be forwarded to a mirror port or the CPU.
3. Although an SPL violation has not actually occurred, the packet should still be forwarded to the CPU. The CPU will determine if an SPL violation has actually occurred. Also, packets with unknown unicast destination addresses, which would normally be flooded to all ports of the switch, will not be flooded to a locked port. A broadcast entry with an address of all ones will be added to the address table so that broadcast packets will be flooded out all ports including source locked ports. Unknown Multicast are always flooded out all ports; only unicast packets are not.

Note that with this function, a per port source address locking register is not necessary. A per port source address locking register is typically used to store the source address of a single node. Only packets containing the corresponding source address would be forwarded. With this port locking function, one address or several address can be locked to a given source port.

5.6.2 Adds

It is often desirable to prevent a user from:

- Attaching to an unused port of a switch
- Disconnecting an existing device on a port, and attaching an unauthorized device
- Attaching to unused ports of a secondary attached device such as a hub

When a packet is received on a locked port and the source address is not found in the address table, it is considered sourced from an unauthorized node. Unknown source addressed packets are sent to the CPU or filtered, as determined by the CPU entry in the Source Port Locking register.

When a given port is locked by setting the corresponding bit in the Source Port locking register, all addresses in the address table with a matching source port field are no longer aged. For these entries, the Don't Age bit in the address table becomes a don't care. When a port is locked the Age bit becomes the Moves bit. This bit determines if a source locked port may be moved to another source locked port. If a packet is received from a sourced locked port and the source address is associated with another sourced locked port and the “Moves” bit for that source address is set, the source port field in the address table for that source address is altered such that the source port field is updated with the new source port. All other fields for the address entry are maintained. In this situation, a Source Port Locking violation has not occurred.

5.6.3 Auto_Source Port Locking

It would be time consuming if each address had to be manually entered into the address table before the locking bits were set. A more hands off use would be to first disable aging on those ports that require the security of the source port locking feature. After enough time has passed to ensure that all addresses have been learned, the port is locked. Only authorized moves and changes require manual configuration.

5.6.4 Static Versus Source Port Locking

Source Port Locking and Static entry functions are not the same.

5.6.4.1 Moves

Moves are not allowed with static entries. Packets are still forwarded as usual regardless of the source port. Static entry moves never result in forwarding a packet to the CPU. If the Age/Moves bit is set to 1, the entry is no longer considered static and moves are allowed. The entry is still not aged.

When a move is detected and the Age/Move bit is cleared, a source port locking violation occurs and the packet is filtered. With source port locking violations, the packet may be forwarded to the CPU. If the Age/Move bit is set, moves are allowed, but only between source locked ports. In this case, a source port locking violation has not occurred and the packet is not forwarded to the CPU.

5.6.4.2 Flooding

Normal flooding is allowed for static entries, however, unknown packets are not flooded to source port locked ports.

5.6.4.3 Learning

The CPU creates static entries. The learning process does not alter static entries. Aging can still occur on ports that have associated static entries. For example, aging still occurs for nonstatic entries in that port addresses are not learned on a port that has been source locked. Aging is not performed on static or source port lock entries.

5.6.4.4 Creation

Static entries are created when the CPU clears the Age/Moves bit and either the Don't Age bit in the address table or the corresponding Don't Age bit in the Port Aging Control Register. All entries with a given source port become source port locked when the Source Port Locking bit for that port is set in the Source Port Locking Register.

5.7 Aging

Address aging is performed automatically. At a programmable interval, the Age/Moves bit of each unicast entry is examined. When the Age/Moves bit is found to be clear, it is set. When source addresses are learned or seen, the ARL clears the Age/Moves bit. If the aging logic discovers an entry with a value of 1, it indicates that there have not been any packets with the corresponding source address processed since the previous interval when the aging logic set the Age/Moves bit. At this point the address is considered to be invalid and is aged from the table and the Empty bit is set. When the empty bit is set, all the other fields of the entry are considered to be invalid and are don't cares.

The aging interval is set according to the value of the Age Interval register. The 20-bit Age Interval register can be programmed for an interval of 5 seconds to 1,000,000 seconds. Every time the Age Interval register expires, each Age/Moves bit in the table is sequentially examined. This means that entries are actually aged anytime between the Age Interval and twice the Age Interval. The default value is 0x3C, or 5 minutes. The default setting results in entries being aged as quickly as 5 minutes and no later than 10 minutes. A value of zero turns automatic aging off. The aging function is the absolute lowest priority access into the address table.

Only the CPU enters or removes Mulicast Entries, therefore there is no aging performed and there is no Don't Age bit. However, the aging logic still examines the entry to determine that it is multicast.

Aging may be disabled for a particular address or on a per port basis under the following conditions:

1. The Don't Age bit is set in the address table entry.
2. The Port Lock bit is set for the source port in the Source Port Lock register.
3. Aging has been disabled for the given port by means of the Port Aging Control register.

5.8 Address Lookup Process

When the arbiter grants a port access, the destination address (DA) and source address (SA) are searched for in the address table. The result of the search is used to generate a forwarding mask.

5.8.1 Unicast

The destination address, extracted by the source port, is fed to all six hash functions. The result is six 11-bit addresses, shifted left one bit to form 12-bit addresses. These addresses are then used in turn to select one of the 2048 hash sites in the address table. At each Hash site there are five possible entries as shown in Table 5.3

Table 5.3 Hash Site Entries

Hash Site Entries	Description
2 Unicast	Both locations contain a unique unicast address.
1 Unicast at the lowest address	Only the first entry contains a unicast address. The second entry is empty.
1 Unicast at the highest address	Only the second entry contains a unicast address. T.he first entry is empty (as a result of aging).
1 Multicast entry	Both entries are used to contain one multicast address.
Empty	Both entries are empty.

As each entry is read from the table, it is compared to the DA for a match. If a match is discovered, the Source Port, Port Mirror and CPU fields are extracted from the entry. (See Figure 5.1.) These fields are used to form the forwarding mask.

If the DA was not found in the address table, an Intermediate forwarding result is created with all fields, except the CPU and Port Mirror, set to 1, indicating that the packet should be flooded to all ports. It must not be forwarded out the port from which it came. Also, if it came from a trunked port, it must not be forwarded out any of the trunk ports. If the source port locking feature has been enabled for a given port, packets are not flooded to that port. Also, if a Mirror port has been defined, the packets are flooded out the mirror port depending upon the setting of the Port Mirror Mask register. If the packet is forwarded out a mirror source port, then it is sent out the mirror port also.

5.8.2 Hashing

The L64324 has a 4096 entry address table. There is a potential for any one of 2^{48} = 281Tera addresses to be in the table. Address hashing is the method of taking an address and mapping it to a single entry in a small address table. The address table is 4096 entries deep, with 2048 Hash sites, requiring an 11-bit index to select a hash site in the table. The function required must take a 2^{48} value and map it to a 2^{11} value. This means that there are $2^{48}/2^{11}$ or 2^{37} potential addresses for each entry in the table. However, there will not be 2^{48} addresses present on the network. Only a few addresses, typically less than a thousand, will

ever be active on the network at any time. So the job at hand is mapping the thousand or so addresses to unique locations within the address table. To increase the probability that this task can be accomplished, multiple hash functions are used to access the table. The functions are applied successively until a match is found. For example, a packet arrives with an address that must be checked against the table. First, hash function #1 is applied to the address. This function yields an 11-bit result that selects one of the 2048 Hash sites in the table. The contents of the locations are compared against the address. If there is a match, the search is successful and ends. If there is not a match, the next hash function is applied as shown below.

```
If A(47:0) = Mem[Hash1(A(47:0))]
   Then Index = Hash1(A(47:0))
 Else if A(47:0) = Mem[Hash2(A(47:0))]
     Then Index = Hash2(A(47:0))
.
.
 Else if A(47:0) = Mem[Hashn(A(47:0))]
    Then Index = Hashn(A(47:0))
```

This continues until the address is found or all hash functions have been exhausted. If an empty location is found at one of the hash sites, this does not mean that the address is new and not in the table yet. This can occur if previous entries have been aged out. Each entry must be checked for a match.

5.8.3 Multicast

The hash function may be as simple as shifting the 48-bit hash functions to the right to include the VLAN_ID and not include the upper 12 bits of the address. Since the upper 3 bytes of the address are always the same for multicast, this is true.

5.9 Learning

The learning process uses the results of the previous search of the source address to determine if the address is new and if there is an empty location. When there is an empty location, an entry is created with the format corresponding to that of a unicast table entry as shown in Figure 5.1, Unicast Address Table entry or as a multicast as shown in

Figure 5.2 Multicast Address Table Entry (First Word) and Figure 5.3 Multicast Address Table Entry (Second Word). New entries are not created when packets are received from source locked ports. Packets from the CPU are treated differently.

5.9.1 Unicast

Unicast entries are created as follows:

Age/Moves
The Age bit is cleared to indicate that it is not ready to be aged.

Don't Age
The Don’t Age bit is cleared to indicate the entry may be aged.

Empty
The Empty bit is cleared, indicating the location is not empty.

CPU
The CPU bit is cleared indicating that packets with this address as a destination should not be forwarded to the CPU.

$\text{Priority}_{\text{CPU}}$
This field should be set to zero.

Port Mirror
The Port Mirror bit is cleared, indicating the address should not be forwarded to the mirror port.

Source Port
This five-bit entry is set to the source port of the packet. Valid entries are 0 to 26, where 0 is the CPU, 1 is port 1, and 25 is the first uplink port.

Address
This is the 48-bit source address.

5.9.2 Multicast

The CPU creates Multicast entries with the format specified in Multicast Entry (Second Word). (Figure 5.3).

5.9.2.1 Moves

When end nodes move between ports on the switch it is necessary to update the node's address table entry to reflect the move. When a source address is found in the table and the source port field does not match the packet's source port, a move has occurred. In this situation the Source Port field for the entry in the address table is updated with the new source port number. The Age/Moves and Empty bits should also be reset at this time.

Setting the corresponding bit in the Source Port Locking register may not move ports that have been locked. The address table entry is not altered and the packet is either discarded or sent to the CPU. Moves are allowed if the Age/Moves bit is set and the move is to another source locked port. In this situation only the source port field is updated.

5.9.2.2 Static Entries

Entries that have the Don't Age bit set in the address table are never aged. Moves are allowed depending on the setting of the Age/Moves bit. If the Age bit is set, Moves are allowed. If the Age/Moves bit is cleared, moves are not allowed. When a move occurs, only the Source port field of the entry is altered All other fields remain the same.

If the corresponding bit is set in the Port Aging Control Register, the entry is never aged, but moves are allowed, depending on the Age/Moves bit. The Port Aging Control Register works in conjunction with the source port locking function.

5.9.2.3 Address Reshuffling

When the last hash function fails to return a matching entry, the address is considered to be new. It is then necessary to learn the address by inserting it in the table. The first step in the process for inserting an address is to use the results of the previous search operation. So while searching for the address initially, it is important to note the first empty hash site. It is in this hash site that the address is stored. If all hash sites are occupied, the address would not be learned (not added) and all traffic with the corresponding address as a destination is flooded to all appropriate ports.

5.9.3 Reshuffling

Consider a new MAC address M. The address needs to be added to the table. However, all six hashes (h1:h6) map to locations that are already occupied by MAC addresses M1:M6. In this event, the CPU can look at the six hash sites that M1 hashes to. If any of these are empty, the CPU can move M1 to the empty location and can thus add M to the location at h1. If the CPU cannot move M1, it tries M2, then M3, and so on. This is a one level reshuffle. If none of M1:M6 can be moved, the CPU can look at the six locations of each of M1:M6 to see if any of those addresses can be moved, thus allowing one of M1:M6 to be moved and allowing the new address to be added. This would be a two level reshuffle. The levels can go on ad infinitum, but there is clearly a lot of CPU work involved as the levels become deeper.

5.9.4 Auto_Reshuffling

Reshuffling has a huge limitation. It requires the CPU to perform the reshuffling, which becomes more CPU intensive with each level of reshuffling. Rather than requiring the CPU to reshuffle the entries, the L64324 lets the learning process perform reshuffling indirectly over time. When an entry fails to be added to the table after all n hashes, rather than just dropping the address and allowing the associated traffic to then flood, one of the 2n hash site addresses is expelled. The expelled entry is replaced by the new address. If and when a packet arrives with the expelled address as the source address, the learning process attempts to add the address back in the table. If the address can find a home within n hashes, the reshuffling is finished. Otherwise, one of 2n addresses is chosen and expelled, and the process repeats again. The hash site chosen is based upon a round robin scheme of the 2n locations.

To drop the address that has failed to be added is to cause those packets targeted at that source to always be flooded. Expelling an address in favor of the new address shifts the flooded packets to traffic associated with the expelled address. The difference is that the expelled address was able to find a home before the new address. The packets sourced from the expelled address will actually not flood, only the first packet will.

Only certain addresses may be expelled from the address table. The following addresses will not be expelled:

- Source locked addresses
- Multicast addresses
- Unicast Addresses with the Don't age bit set
- Unicast Addresses with the CPU bit set
- Unicast Addresses with the Port Mirror Bit set
- Addresses where aging has been disabled for the source port, as indicated in the Port Aging Control Register

All other addresses can and will be expelled. This may be considered as early aging. This list may be reduced down to just multicast and addresses with the Don’t age bit set. Since the Don’t age bit should really be set for the other cases as well.

The Address Resolution Logic is responsible for adding, modifying and finding address table entries on behalf of the CPU. The CPU has the lowest priority of service next to Aging. However, since the Address Resolution Logic is very fast there is more than enough bandwidth to go around.

The Interface supports the address table maintenance commands as defined in the ADDR_CMD Register (address 0x3425.0004). Commands are issued by means of the address and command registers in conjunction with four other registers.

Chapter 6
Port Trunking

This chapter provides a description of the L64324 24 x 2 Fast Ethernet Intelligent Switch Port Trunking and contains the following sections:

6.1 Overview

Port trunking is accomplished when ports (links) are grouped into a trunk, thereby increasing available bandwidth. Instead of having one high-bandwidth link between segments, you can combine two or more lower bandwidth links to improve the throughput between segments. Port trunking also provides fault tolerance. If one link fails, one of the other links continues to deliver packets to the addresses assigned to the port connected to the failed link. Port trunking works on essentially an all-or-none principle. As long as one link is up, the trunk is up. If all the links are down, the trunk is down.

The L64324 port trunking has the following characteristics:

- Supports trunking based on SA and DA.
- Supports SA only trunking for legacy systems.
- Provides adequate load balancing with minimal logic.
- Handles failover automatically, with the link down condition discovered in hardware.
- Prevents out of sequence traffic during failover and link up.
- Localizes functions to minimize switch system impact.
- Delivers packets to CPU with actual port number information.
- Delivers trunking on any four ports.
- Aggregates two, three, or four ports.
- Supports one trunk only.
- Initialization, configuration, and link addition controlled through software.
- Uses an additional MAC address to support the aggregation protocol.
- Provides software override of hardware functions.

6.2 Frame Forwarding Methods

When two or more ports have been combined to form one logical higher bandwidth link, a method is needed to determine how the traffic is sent across each individual link. For example, if there is a packet ready to cross the trunk, which of the links that form the trunk should be used to transfer the packet? There are several ways to make this decision. At the data link layer, this decision can be made based on the DA, SA, or a combination of source and destination addresses.

6.2.1 DA Frame Forwarding

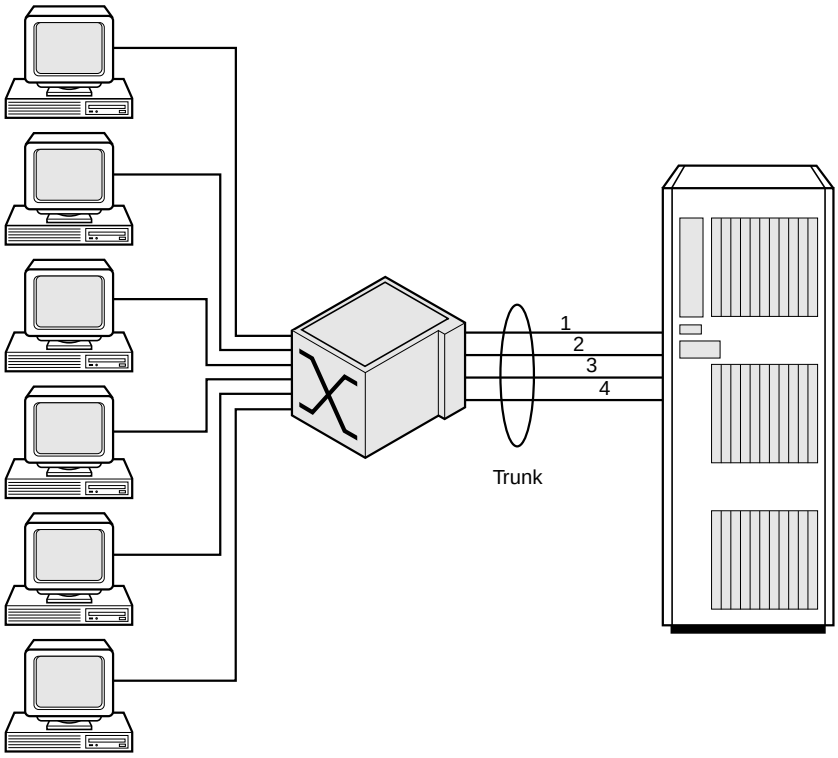
With DA frame forwarding, packets destined for the trunk are forwarded just like any other packet. The destination address is used to index the address table and the result is a destination port number that is used as the forwarding port. The difference between a trunked port and a

nontrunked port is the way in which the source addresses are learned. Initially, when packets are flooded across the trunk, each new source address seen is assigned to one of the ports that forms the trunk. This assignment is independent of the actual port on which the packet is received. The assignment mechanism used to select the port is chosen so that it distributes the learned source addresses evenly across the links of the trunk. After a source address has been learned and assigned to a given port, it remains assigned to that port until it is aged out.

Note : The source address is not necessarily always received on the port to which it has been assigned.

Again, the assignment is not related to the port on which the source address was originally seen. To support this structure, source addresses learned must be known as sourcing from across the trunk. This prevents the learning mechanism from reassigning a physical port to the source address each time the source address is received on a different trunk link. When the learning mechanism examines a source address that has been received from the trunk, and the source port is that of a trunk link, the physical port that it was actually received on has no importance, and the source port for the address table entry is not altered. The limitation of this forwarding method is that traffic to a given destination always crosses the same physical link. This is especially limiting in a client-server environment where all the clients are communicating with the same server(s), as shown in the example shown in Figure 6.1.

Figure 6.1 DA Only Port Trunking Example



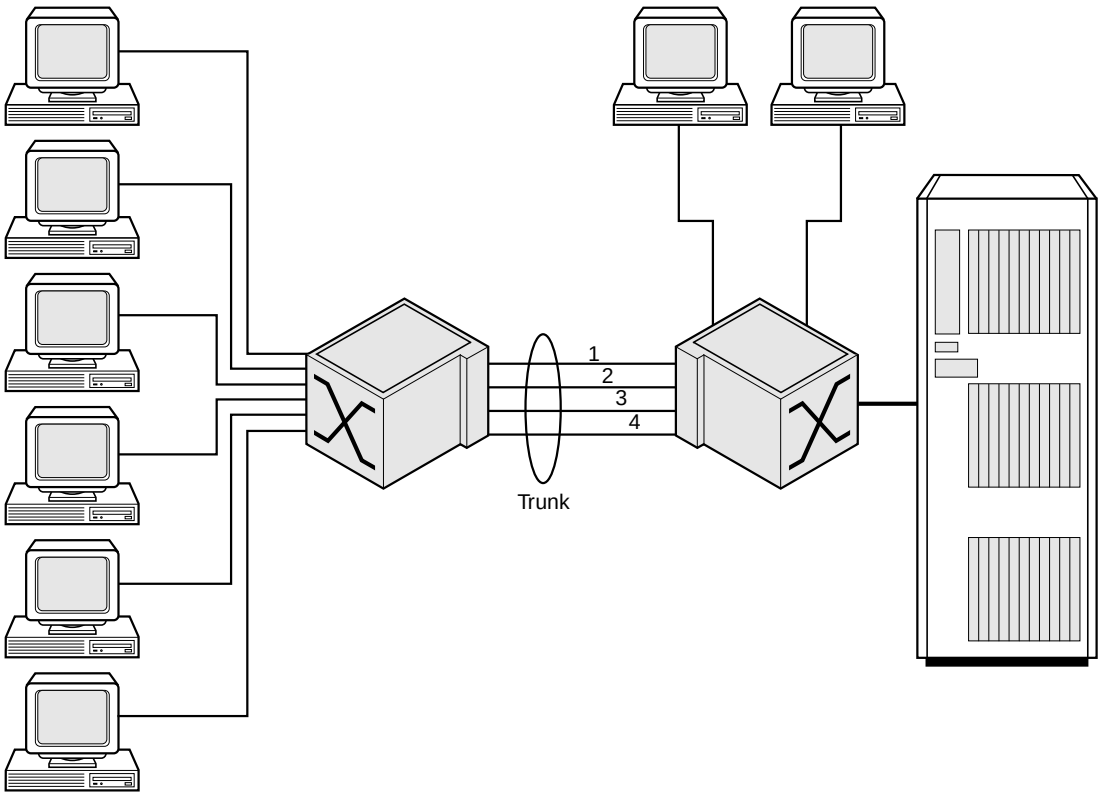
6.2.2 SA Frame Forwarding

In SA frame forwarding, the link chosen is based directly on the source address of the packet. Although the link is chosen based on the source address alone, the destination address is still used to determine that the packet is destined for a trunked link. For example, when a packet is received from a nontrunked port, the destination address is looked up in the address table to determine how to forward the packet. If the destination address is associated with one of the trunk links, the source address is then used to actually determine on which of the trunked links the packet is forwarded. Similar to DA only forwarding, an assignment process attempts to distribute the learned source addresses uniformly across the links of the trunk. Once assigned, packets destined for the trunk from a given source always use the same physical link of the trunk. Also similar to DA forwarding, the link on which a packet is forwarded is not necessarily related to the link the return traffic takes.

In the example of Figure 6.2, each of the machines (the nodes and the server) has a single unique MAC address, which is used as the SA when packets originate from the respective machine.

The traffic destined to the server originates from different end nodes, which each contain a different SA. The result is that when the trunking is based on SA only, the traffic from the end nodes that traverses the two switches in the diagram is distributed among the four physical links according to the SA.

Figure 6.2 SA Only Port Trunking Example



All of the traffic originating from the server and destined for the end nodes contains the same SA (that of the server), irrespective of the destination node. So when trunking is based on SA only, the traffic from the server ends up using only one link of the trunk between the switches shown in the diagram. The remaining three links are relatively unused.

6.2.3 SA/DA Frame Forwarding

With either SA or DA frame forwarding, traffic is poorly distributed across the links of the trunk in one direction or the other when there are traffic environments of many to one or one to many. To avoid this issue, SA/DA forwarding forwards traffic based on both the source and destination addresses simultaneously. This allows the physical port forwarding selection to be made on a per conversation basis. Table 6.1 is an example of SA/DA forwarding. For each combination of source and destination address, one of four possible trunk links has been assigned (designated with the numbers 1, 2, 3, or 4).

Table 6.1 SA/DA Port Trunking Forwarding Table

Source Addresses	Destination Addresses			
	080090123434	**08009011145**	**080090122256**	**080090126546**
080090123455	1	2	3	4
080090123422	3	2	4	1
080090123411	2	1	1	3
080090123006	3	4	4	4

When a packet arrives, its source and destination addresses are used to index the table to determine on which port the packet will be transmitted. This approach allows traffic with any given source or destination address to cross any of the links of the trunk. For example, a packet that contains the SA corresponding to the first entry in the table can be sent across any of the four possible links, depending on which of the destination addresses the packet contains. SA/DA forwarding provides good traffic distribution across the links of the trunk, independent of traffic direction or traffic environment.

It is important to note that if all the links of the trunk were monitored, a given source or destination address may be seen crossing any or all the links at any given time.

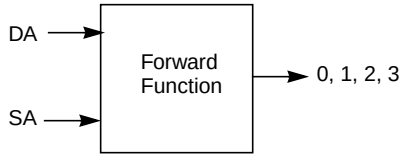
In practice, such a table would consume a great deal of logic, and would be difficult and slow to access and maintain. In addition, it is not necessary to provide absolute control of each conversation pertaining to

the physical trunked port that forwards the packet. It is only necessary to adequately distribute the possible conversations evenly across the trunked links. For this reason, only a portion of the SA or DA, or a function that acts to deliver a modified portion is necessary. Note that this mechanism does not provide true load balancing. Ideally, monitoring the bandwidth that each conversation consumes or the bandwidth associated with a given address can be used to “load balance” traffic across links of a trunk. However, such a scheme would be overkill for this application. There is no standard that provides a method for distribution. The method chosen is implementation dependent. The SA/DA forwarding function is described in Section 6.3, “Port Trunking Logic Forwarding Function.”

6.3 Port Trunking Logic Forwarding Function

There are two rather simple SA/DA schemes that can be used to select among the trunked ports. First, a simple table-based lookup can be used. This scheme takes only a portion of the SA and DA to access a 16-entry table. Software then programs the contents of the table. Each entry in the table corresponds to a trunked link. The alternate method is to use a fixed logical function which again uses only a portion of both the SA and DA. The output of the logical function is a logical trunked port number, as shown in Figure 6.3.

Figure 6.3 Port Trunking Logic



The forwarding logic function must generate four possible outcomes for any particular DA or SA, because there are only four possible links in the trunk. The four outcomes are then used to select one of four logical trunk ports (see Section 6.10, “LMPR Register,” page 6-23 for more details). The entire DA and SA could be supplied to the logic function, but there is no need for this level of exactness. The simplest logic function would be to take the least-significant bit of both the SA and DA and use the result directly. For example, a DA ending in 0x03 (0b0011) and an SA

ending in 0x08 (0b1000) gives a port result of 0b10, or logical port 2, with the DA LSB being the MSB of the port result and the SA LSB being the LSB of the result. The drawback of this function is that for any given address, the traffic always crosses over only two of the four links. As shown in Table 6.2, a given source address always falls in a single column and a destination address always falls in a given row with only two possible outcomes.

Table 6.2 Simple Logic Function

	SA LSB	
DA LSB	0	1
0	00	01
1	10	11

This simple logic function clearly demonstrates that the desired function must allow any given SA or DA to use any of the four links. This means the function must result in a table that has four outcomes for a given DA input and four outcomes for a given SA. Table 6.3 indicates such a function.

Table 6.3 Trunking Four-Outcome Table

		SA (2 LSBs)			
Function = f(DA[1:0], SA[1:0])		00	01	10	11
DA (2 LSBs)	00	0	1	2	3
	01	1	0	3	2
	10	2	3	0	1
	11	3	2	1	0

As shown in Table 6.3, a given source address corresponds to the destination addresses in a single column. The SA can now result in any of the four potential trunk links, depending on the DA it is paired with. Likewise, a given DA corresponds to the SA in a single row. The DA can result in any of the four potential trunked links, depending on the SA.

Note that the entry assignments in the table are chosen such that for any DA or SA, all four links are possible.

6.4 MAC Address Assignment Compensation

Before we examine the function that describes the table, notice that there are now four bits of input required. The information used to index Table 6.4 is some representation of the SA and DA. Statistically, network nodes from different manufacturers have a unique lower three bytes. Network nodes from the same manufacturer could have sequential addresses. Typically, the least-significant bits of the address are chosen as an index because they are the most likely to be variable from node to node. However, not all vendors have chosen to increment or decrement the least-significant field of the MAC address in order to assign MAC addresses. As shown in Table 6.4, a MAC address contains two fields:

- Manufacturer
- Manufacturer assigned

Table 6.4 MAC Address Fields

Manufacturer	Manufacturer Assigned
XX-XX-XX	XX-XX-XX

The Manufacturer field is assigned in accordance with IEEE STD 802-1990.

The manufacturer directly administers the Manufacturer Assigned portion of the 48-bit MAC address. A given manufacturer may assign the 16.8 million possible MAC addresses in any sequence necessary. The Manufacturer field is essentially constant and only changes when a manufacturer allocates all 16.8 million addresses. The Manufacturer Assigned field contains the three bytes as indicated in Table 6.4. It is probable that the manufacturer will assign the addresses in some logical incrementing or decrementing sequence.

However, some manufacturers have held the least-significant bytes constant and sequenced the most significant byte(s) to assign addresses. For this reason, the two least-significant bits from each byte

of the Manufacturer Field are logically combined to form the index. The combination is such that if the bytes are constant, they will not have an effect on the result from address to address. For example, if the corresponding bits of each byte are XOR'ed, the address variation is captured, as indicated in the three example addresses shown in Table 6.5. In this example, it is the second byte that is sequenced, yet the variation is captured.

Table 6.5 MAC Address Assignment Compensation

MAC Addresses (Lower Half)		
34-01-65	34-02-65	34-03-65
0011-0100-0000-0001-0110-0101	0011-0100-0000-0010-0110-0101	0011-0100-0000-0011-0110-0101
$IndexSA_0 = SA_0 \oplus SA_8 \oplus SA_{16}$ $IndexSA_1 = SA_1 \oplus SA_9 \oplus SA_{17}$ For $IndexSA_0$: $1 \oplus 1 \oplus 0 = 0$ For $IndexSA_1$: $0 \oplus 0 \oplus 0 = 0$ Therefore, SA[1:0] = 00	$IndexSA_0 = SA_0 \oplus SA_8 \oplus SA_{16}$ $IndexSA_1 = SA_1 \oplus SA_9 \oplus SA_{17}$ For $IndexSA_0$: $1 \oplus 0 \oplus 0 = 1$ For $IndexSA_1$: $0 \oplus 1 \oplus 0 = 1$ Therefore, SA[1:0] = 11	$IndexSA_0 = SA_0 \oplus SA_8 \oplus SA_{16}$ $IndexSA_1 = SA_1 \oplus SA_9 \oplus SA_{17}$ For $IndexSA_0$: $1 \oplus 1 \oplus 0 = 0$ For $IndexSA_1$: $0 \oplus 1 \oplus 0 = 1$ So, SA[1:0] = 10

If only the lower byte was XOR'ed, the same result would have been returned for all three addresses. Independent of the manufacturer, or the byte that was chosen to be sequenced, this operation provides a reasonable index for the port trunking logic function.

In summary, the index is obtained as follows:

$IndexSA_0 = SA_0 \oplus SA_8 \oplus SA_{16}$

$IndexSA_1 = SA_1 \oplus SA_9 \oplus SA_{17}$

$IndexDA_0 = DA_0 \oplus DA_8 \oplus DA_{16}$

$IndexDA_1 = DA_1 \oplus DA_9 \oplus DA_{17}$

6.5 Port Trunking Forwarding

As indicated in the section above, the least-significant two bits of each byte are XOR'ed in order to generate an index that is fed to the function or table to generate the port trunk number. The function selected assigns the entries in Table 6.3. The important characteristic of the function is that it must result in all four port combinations in each row and in each column. For the four-by-four matrix of Table 6.3, a simple XOR function provides the desired result. The same significant bit of address index is XOR'ed with the other address index.

6.5.1 XOR Port Trunking Forwarding Function

Logical port = $[LP_1, LP_0]$

$LP_0 = IndexSA_0 \oplus IndexDA_0$

$LP1 = IndexSA_1 \oplus IndexDA_1$

For example, if the index for the DA is $IndexDA_{0,1}$ = 0b01, and if the index for the SA is $IndexSA_{0,1}$ = 0b11**,** the results are summarized in Table 6.6.

Table 6.6 XOR Trunk Function

Index	Value
$IndexDA_{0,1}$	0b01
$IndexSA_{0,1}$	0b11
Result (Logical Port)	0b10 (2)

Hence, the combination of the SA and DA indexes shown in Table 6.6 would result in the packet being sent on logical trunk port 2. Using this function, the result is as shown in Table 6.7.

Table 6.7 Basic XOR PT Forwarding Table

		SA			
Function = XOR		00	01	10	11
DA	00	0	1	2	3
	01	1	0	3	2
	10	2	3	0	1
	11	3	2	1	0

Entries in the table correspond to logical trunks (see Section 6.10, “LMPR Register,” page 6-23).

The XOR function is the default or base function used when there are two- or four-trunk ports. When there are only two ports, only the least significant result of the XOR need be used and is represented by the entries in the grayed area of Table 6.7.

6.6 Three Link Trunk

When there are three links in a trunk, the binary function of Table 6.7 is not as easily applied. For example, if there are three logical ports, which port should be used when the DA and SA index selects logical port 3, which does not exist? This situation is illustrated in Table 6.8

Table 6.8 Only Three Trunk Links (Port 3 Not Assigned)

		SA			
Function = ?		00	01	10	11
DA	00	0	1	2	?
	01	1	0	?	2
	10	2	?	0	1
	11	?	2	1	0

The entries that are made in place of the question marks in Table 6.8 are made to allow a given SA or DA to be evenly distributed across the three links. The problem is that the attempt to select among the three possibilities is based on a power of two mechanism. For this reason, a less than perfect assignment is made, as shown in Table 6.9.

Table 6.9 Only Three Trunk Links (Port 3 Reassigned)

		SA			
Function = ?		00	01	10	11
DA	00	0	1	2	2
	01	1	0	1	2
	10	2	0	0	1
	11	2	2	1	0

The assignments indicated in Table 6.9 might at first seem adequate, because there is a fairly even distribution among the three logic ports. The weakness is that given a many-to-one traffic environment, the distribution becomes 50/25/25. This is clearly shown as you examine the entries in any column or row of Table 6.9. In each case, there is one logical port listed twice, resulting in a 50% allocation for the port. One way to correct this problem is to enlarge the table to have 8 or 16 entries in a column. Because the L64324 is primarily a desktop switch, with traffic flows of many to few, it is important to prevent an uneven distribution based on destination address. For this reason, and to simplify the logic, only the width of the table is increased. This prevents traffic to a given server from oversubscribing a particular link.

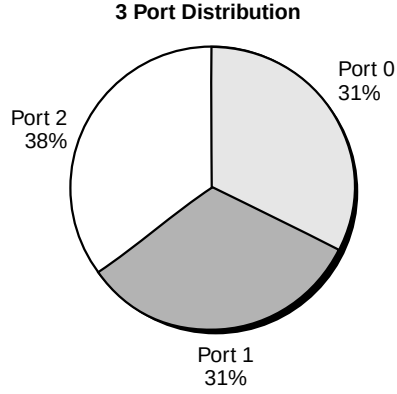
Table 6.10 shows that for any destination address, there is a statistical distribution of 25/37.5./37.5% among the three ports. Increasing the table width to 16 entries would improve the statistical distribution to 37.5/31.25/31.25%.

Table 6.10 Only Three Trunks (Eight Entries For Improved Distribution)

Function = ?		SA							
		000	001	010	011	100	101	110	111
DA	000	0	1	2	2	0	1	2	1
	001	1	0	1	2	1	0	0	2
	010	2	0	0	1	2	2	0	1
	011	2	2	1	0	1	2	1	0

The benefit of 16 entries over 8 as shown in Figure 6.4 is clear, but questionable. However, because the implementation requires only an additional 200 to 500 gates, the cost difference is insignificant; hence, the 16-entry table width is used.

Figure 6.4 16-Entry Trunking Table Distribution



6.7 Trunking Combinations

When a port fails, distributing the SA or DA evenly across the remaining links complicates the failover functions. Depending on which port fails, the logical port table is filled in differently. However, in the case of a four-port logical trunk, all the tables begin with the base table shown in .

Table 6.11 Four-Port Logical Trunk Base Table

All Ports Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
01	1	0	3	2	1	0	3	2	1	0	3	2	1	0	3	2
10	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1
11	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0

The tables that follow show how the selected port in a trunk depends on the SA/DA index and which ports are failed or alive out of all the trunked ports.

The tables are also optimized so that the best fair distribution is obtained. When all ports are alive, 25% of the traffic exists on each port (assuming traffic that generates all the SA/DA index combinations fairly). Under a three-port trunking case, and when the SA/DA indexes are used, traffic is distributed as 34.375%, 32.8125%, and 32.8125% (one port has 34.375% of the traffic while the other two have 32.8125%).

Under the same three-port trunking condition, if only the SA index is used, the traffic distribution is 37.5%, 31.25%, and 31.25% Under a two-port trunking condition, the traffic distribution is 50% and 50%.

Care is taken to ensure that when a link fails, the existing traffic on live ports is not redistributed. Only traffic that was going to the failed port is redistributed. For example, the tables that show ports 1, 2, and 3 alive and 2 and 3 alive differ only in the positions where port 1 existed in the table where ports 1, 2, and 3 were alive.

Note that in the following tables, a number in a table cell that is not shaded indicates a live port number, a shaded table cell indicates a failed port, and a number within a shaded cell indicates the failed-over port number. The shading usage is:

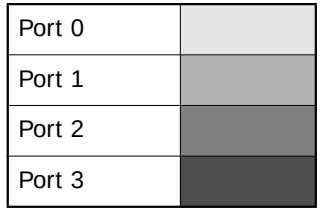


Table 6.12 Ports 1, 2, and 3 Alive, Port 0 Failed

1, 2, 3 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1	1	2	3	1	1	2	3	3	1	2	3	2	1	2	3
01	1	2	3	2	1	3	3	2	1	2	3	2	1	3	3	2
10	2	3	3	1	2	3	2	1	2	3	1	1	2	3	1	1
11	3	2	1	1	3	2	1	1	3	2	1	2	3	2	1	3

Table 6.13 Ports 0, 1, and 3 Alive, Port 2 Failed

0, 1, 3 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0	1	3	3	0	1	3	3	0	1	1	3	0	1	0	3
01	1	0	3	1	1	0	3	1	1	0	3	0	1	0	3	3
10	3	3	0	1	1	3	0	1	0	3	1	1	3	3	0	1
11	3	0	1	0	3	0	1	0	3	1	1	0	3	1	1	0

Table 6.14 Ports 0, 1, and 2 Alive, Port 3 Failed

0, 1, 2 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0	1	2	0	0	1	2	0	0	1	2	1	0	1	2	2
01	1	0	2	2	1	0	2	2	1	0	0	2	1	0	1	2
10	2	1	0	1	2	1	0	1	2	2	1	1	2	0	0	1
11	0	2	1	0	0	2	1	0	1	2	0	0	2	2	1	0

Table 6.15 Ports 0, 2, and 3 Alive, Port 1 Failed

0, 2, 3 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0	3	2	3	0	3	2	3	0	0	2	3	0	2	2	3
01	2	0	3	2	2	0	3	2	3	0	3	2	0	0	3	2
10	2	3	0	2	2	3	0	2	2	3	0	3	2	3	0	0
11	3	2	0	0	3	2	0	0	3	2	2	0	3	2	3	0

Table 6.16 Ports 1 and 3 Alive, Ports 0 and 2 Failed

1, 3 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1	1	3	3	1	1	3	3	3	1	1	3	3	1	1	3
01	1	1	3	1	1	3	3	1	1	1	3	3	1	3	3	3
10	3	3	3	1	1	3	1	1	3	3	1	1	3	3	1	1
11	3	3	1	1	3	3	1	1	3	1	1	3	3	1	1	3

Table 6.17 Ports 1 and 2 Alive, Ports 0 and 3 Failed

1, 2 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1	1	2	2	1	1	2	1	2	1	2	1	2	1	2	2
01	1	2	2	2	1	2	2	2	2	2	1	2	1	1	1	2
10	2	1	2	1	2	1	2	1	2	2	1	1	2	2	1	1
11	1	2	1	1	2	2	1	1	1	2	1	2	2	2	1	2

Table 6.18 Ports 2 and 3 Alive, Ports 0 and 1 Failed

2, 3 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	2	3	2	3	3	3	2	3	3	2	2	3	2	2	2	3
01	2	2	3	2	2	3	3	2	3	2	3	2	3	3	3	2
10	2	3	3	2	3	2	2	2	3	2	3	3	2	3	2	3
11	3	2	3	2	3	2	2	3	3	2	2	2	3	2	3	3

Table 6.19 Ports 0 and 3 Alive, Ports 1 and 2 Failed

0, 3 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0	0	3	3	0	3	3	3	0	0	0	3	0	0	0	3
01	0	0	3	3	0	0	3	3	3	0	3	0	0	0	3	3
10	3	3	0	0	3	3	0	0	0	3	0	3	3	3	0	0
11	3	0	0	0	3	0	0	0	3	3	3	0	3	3	3	0

Table 6.20 Ports 0 and 2 Alive, Ports 1 and 3 Failed

0, 2 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0	0	2	0	0	2	2	0	0	0	2	2	0	2	2	2
01	2	0	2	2	2	0	2	2	0	0	0	2	0	0	0	2
10	2	2	0	2	2	0	0	2	2	2	0	0	2	0	0	0
11	0	2	0	0	0	2	0	0	2	2	2	0	2	2	2	0

Table 6.21 Ports 0 and 1 Alive, Ports 2 and 3 Failed

0, 1 Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0	1	1	0	0	1	0	0	0	1	1	1	0	1	0	1
01	1	0	0	1	1	0	1	1	1	0	0	0	1	0	1	0
10	1	1	0	1	1	1	0	1	0	0	0	1	0	0	0	1
11	0	0	1	0	0	0	1	0	1	1	1	0	1	1	1	0

Table 6.22 Port 0 Only Alive, Ports 1, 2, and 3 Failed

0 Only Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6.23 Port 1 Only Alive, Ports 0, 2, and 3 Failed

1 Only Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
01	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
11	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 6.24 Port 2 Only Alive, Ports 0, 1, and 3 Failed

2 Only Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
01	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
10	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
11	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

Table 6.25 Port 3 Only Alive, Ports 0, 1, and 2 Failed

3 Only Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
01	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
10	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
11	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

Table 6.26 No Ports Alive

None Alive	Source Address Index															
Destination Address Index	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
01	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
10	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
11	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

6.8 New Link Active (Software Controlled)

It is the responsibility of software to manage and control the creation of the trunk. Software is also responsible for adding ports to an existing trunk. Managing the Logical to Physical Map Register (LMPR), port Spanning Tree Protocol (STP) bits, and Trunk Method accomplishes this initialization and control. All other details are handled in hardware (although it is the responsibility of the software to make sure the associated control setting for Source Port Locking, Port Mirroring, and entries in the address table associated with the port are set consistently with the other trunked links). When a port is added to the trunk, the appropriate LMPR[3:0] field is programmed accordingly. The learning engine then uses this information to determine if entries in the address table were sourced from a trunk port. If packets are destined to a port that is programmed into one of the LMPR[3:0] fields, the trunk forwarding logic is used to determine which of the trunk links is used to forward the packet.

To form a trunk, follow these steps:

1. Set the Trunking Method (SA/DA or SA)
2. Disable the target ports (STPstate = Disabled)
3. Wait for the associated queues of the disabled ports to be emptied. (Read = Write pointer)
4. Program the LMPR.
5. Set the STP state of the ports to Forward.

To add a link to a trunk, follow these steps:

1. Disable the current ports of the trunk. (STPstate = Disabled)
2. Wait for the associated queues of the disabled ports to be emptied. (Read = Write pointer)
3. Program the LMPR with the additional port.
4. Set the STP state of the ports to Forward.

To remove a link from a trunk, follow these steps:

1. Disable the port. (STPstate = Disabled)
2. Set the corresponding LMPR bit to zero

   Hardware handles the rest

To remove a trunk, set the LMPR to zero

6.9 Source Address Forwarding

Setting the TRUNK_METHOD bit in the LMPR controls the forwarding method. At power-up the LMPR is set to zero. The SA/DA method is selected by default when the TRUNK_METHOD bit is 0. When the TRUNK_METHOD bit is 1, SA only trunk forwarding is selected. When SA forwarding is selected, the packet is forwarded based only on the source address. Specifically, the process is the same as DA/SA forwarding except that the DA index is always 00 (Index$DA_{0,1}$= 0b00**)**, as shown in Table 6.27.

Table 6.27 Three Port Logical SA Only Trunk Table, Logical Port 3 Failed or Not Present

		SA Index															
Function = XOR		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
DA	00	0	1	2	0	0	1	2	0	0	1	2	1	0	1	2	2

6.10 LMPR Register

As shown in Table 6.28, there are four Logical-to-Physical port Mapping register fields, LPMR[3:0].

Table 6.28 LMPR Register, ADDR = 0x3425.0048

Bit	Name	R/W	Default	Description
31:30	Reserved	–	–	–
29	Trunk_Method	R/W	0	0 = SA/DA trunking 1 = SA only trunking
28:20	Reserved	–	–	–
19:15	LMPR3	R/W	0	Set the physical port for the logical trunk link 3. A value of 0 removes the logical port from the trunk 0 <= LMPR3 <= 26
14:10	LMPR2	R/W	0	Set the physical port for the logical trunk link 2. A value of 0 removes the logical port from the trunk 0 <= LMPR2 <= 26
9:5	LMPR1	R/W	0	Set the physical port for the logical trunk link 1. A value of 0 removes the logical port from the trunk 0 <= LMPR1 <= 26
4:0	LMPR0	R/W	0	Set the physical port for the logical trunk link 0. A value of 0 removes the logical port from the trunk 0 <= LMPR0 <= 26

These R/W fields map the logical trunking ports to the physical trunking ports. Because the L64324 supports only a single port, there are only four logical trunking ports, and a single trunk can contain from one to four ports.

As shown in Table 6.29, each LPMR contains a 5-bit port number that corresponds to a physical port. The internal trunking logic maps each packet to one of four logical ports depending on the source address or source/destination address pairs.

Table 6.29 LMPR Registers

Logical Port	Physical Port	Register
0	01100	LMPR0
1	00001	LMPR1
2	01011	LMPR2
3	00110	LMPR3

It is the responsibility of software to initialize these registers with the physical port numbers. Once initialized, the hardware handles failover conditions automatically. See Section 6.7, “Trunking Combinations,” page 6-14 for more details. The range of possible port numbers is 1–26 (0b00001–0b11010). These registers are cleared to 0 at power up. An entry of 0 indicates that a trunk has not been initialized. All entries must be either zero or nonzero.

Table 6.30 No Trunk

Logical Port	Physical Port	Register
0	0	LMPR0
1	0	LMPR1
2	0	LMPR2
3	0	LMPR3

One trunk is a valid configuration, but with little value:

Table 6.31 LMPR, One Trunk Link

Logical Port	Physical Port	Register
0	6	LMPR0
1	6	LMPR1
2	6	LMPR2
3	6	LMPR3

Mapping the four logical ports to two physical ports forms a two-link trunk, as shown in Table 6.32.

Table 6.32 LMPR, Two Link Trunk

Logical Port	Physical Port	Register
0	1	LMPR0
1	6	LMPR1
2	1	LMPR2
3	6	LMPR3

A full four link trunk is configured by loading all four LMPR with unique port numbers, as shown in Table 6.33.

Table 6.33 LMPR, Four Link Trunk

Logical Port	Physical Port	Register
0	1	LMPR0
1	6	LMPR1
2	2	LMPR2
3	7	LMPR3

When a failover condition occurs or when there are only three, two, or one remaining active port, the hardware automatically selects the appropriate LMPR(s). This logic maps all traffic to the logical trunk ports that correspond to active links. This logic yields a statistical three port distribution of 37.5%, 31.25%, and 31.25% among the logical ports.

Table 6.34 LMPR With Only Three Links in a Trunk

	Physical Ports	Port 1 Inactive[1]	Port 4 Inactive	Port 2 Inactive	Port 1 Inactive
LMPR0	6	6	6	6	X
LMPR1	8	8	8	X	8
LMPR2	4	4	X	4	4
LMPR3	1	X	1	1	1

1. Default for 3-port trunk initialization.

Table 6.35 LMPR With Only Two Links in a Trunk

	Physical Ports	Port 4 Inactive[1]	Port 8 Inactive	Port 6 Inactive
LMPR0	6	6	6	X
LMPR1	8	8	X	8
LMPR2	4	X	4	4
LMPR3	0	0	0	0

1. Default for 2-port trunk initialization.

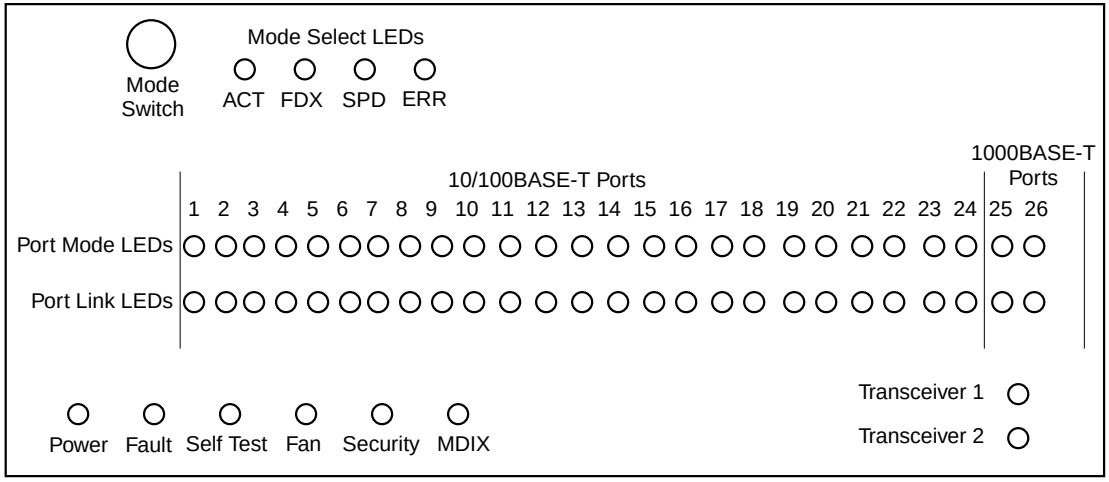
Chapter 7
LED Control

This chapter describes the LED functions of the L64324 Fast Ethernet Intelligent Switch. Some LEDs are controlled directly from hardware, and some can be controlled either from hardware or software. The chapter contains the following sections:

7.1 Introduction

The L64324 drives external LEDs through external 8-bit Serial to Parallel Shift Registers. It is relatively simple to organize the LEDs so that they are presented on a front panel with clear labeling, which allows easy and straightforward monitoring of device functions and operation. An example of a LED panel for the device is shown in Figure 7.1.

Figure 7.1 Sample LED Panel



All LEDs can be directly controlled from the CPU or from one of two global programmable timers. Most LEDs can also be directly controlled from one of the four hardware functions. All LEDs illuminate for several seconds after power up or reset.

7.2 Mode Switch and Mode Select LEDs

The Mode Switch selects one of four display modes:

- ACT: Activity
- FDX: Full/Half Duplex
- SPD: 10/100 Mbits/s Speed
- ERR: Error

There are four Mode Select LEDs shown in Figure 7.1 They are arranged in the following sequence from left to right: ACT, FDX, SPD and, ERR. Only one mode is active at any time. Therefore only one of the four Mode Select LEDs should be illuminated during normal operating conditions. The default mode is ACT (Activity).

Each time the mode switch is depressed, the current Mode Select LED is turned off and the Mode Select LED to the right of the current LED is illuminated. The CPU controls the Mode Select LEDs in response to depressing the mode switch.

When a mode is selected, the individual Port Mode LEDs display the function selected. For example, if the ACT Mode Select LED is on, the Port Mode LEDs for each port reflect the activity state of that port.

7.3 Port Mode LEDs

Each of the Port Mode LEDs (one per port) reflects the operation of the port corresponding to the particular Mode Select LED that is illuminated, as summarized in Table 7.1.

Table 7.1 Port Mode LEDs Operation

Mode Select LED Illuminated	Port Mode LED State	Description
ACT	Off	No network traffic exists on the given port, or the port is disabled
	On	Network traffic exists on the given port
FDX	Off	Port is operating in Half Duplex.
	On	Port is operating in Full Duplex
SPD	Off	Port is operating at lower speed
	On	Port is operating at higher speed
ERR	Off	No errors occurring on the port
	On	Any error detected illuminates the LED for 5 ms

7.3.1 Activity Mode (ACT)

Activity is defined as sending or receiving any good or bad packet. When activity occurs on a port, and if the ACT Mode Select LED is on, the Port Mode LED for that port illuminates for a duration of 5 ms. The 5 ms is a fixed time interval; any activity occurring during the 5 ms does

not lengthen the interval past 5 ms. Immediately after the 5 ms duration, the Port Mode LED is extinguished. The LED then lights for 5 ms for the next activity event. There is no specified off time.

Besides packet transmission and reception, pause frame transmission and reception is also an activity event. When the hardware control multiplexer (See Section 7.7.1, “Port Mode and Link LEDs Hardware Multiplexer”) is configured to source the LEDs with activity events, all Port Mode LEDs are sourced with activity events. A disabled port should not reflect receive activity.

7.3.2 Full Duplex Mode (FDX)

When a given port has been configured for or has negotiated to Full Duplex operation, and if the FDX Mode Select LED is on, the Port Mode LED for that port illuminates. When the hardware control multiplexer (See Section 7.7.1, “Port Mode and Link LEDs Hardware Multiplexer”) is configured to source the LEDs with the duplex state, all Port Mode LEDs are sourced with the duplex state.

7.3.3 Speed Mode (SPD)

When the hardware control multiplexer (See Section 7.7.1, “Port Mode and Link LEDs Hardware Multiplexer”) is configured to source the LEDs with speed state, all Port Mode LEDs are sourced with the speed state. A Port Mode LED illuminates if the given port is operating at its fastest speed. Port Mode LEDs for the 10/100 ports light only when the port has been configured for or has negotiated to 100 Mbits/s operation. Port Mode LEDs for the 100/1000 ports light only when the port has been configured or has AutoNegotiated to 1000 Mbits/s operation.

7.3.4 Error Mode (Err)

When an error event occurs, and if the ERR Mode Select LED is on, the Port Mode LED for that port illuminates for a duration of 5 ms. The 5 ms is a fixed time interval; any error event occurring during the 5 ms does not lengthen the interval past 5 ms. Immediately after the 5 ms duration, the Port Mode LED is extinguished. The LED then lights for 5 ms for the next error event. There is no specified off time.

When the hardware control multiplexer (See Section 7.7.1, “Port Mode and Link LEDs Hardware Multiplexer”.) is configured to source the LEDs with error events, all Port Mode LEDs are sourced with error events.

7.4 MDIX Port Mode

MDIX corresponds to the default or standard connection configuration of a switch port. MDI corresponds to the standard configuration of an end port node. In this mode of operation, the hardware control multiplexer is not important, because the CPU controls this mode. The CPU continually polls the PHYs to determine the MDI state. If a port is found to be in the MDI state, the corresponding bit in the LED registers is set.

It is not likely nor is it the intent of this mode to reflect the state of the MDI mode of each interface.

7.5 Port Link LEDs

There are 26 Port Link LEDS. Their operation is as summarized in Table 7.2.

Table 7.2 Port Link LED Operation

Link LED State	Description
Off	One of these conditions exists: • a network cable is not connected to the port • the port is not receiving link beat or sufficient light energy • the port has been disabled
On	Indicates the port is enabled and receiving a link beat signal (for the twisted-pair ports), or a strong enough light level (for the fiber-optic ports on the switch modules) from the connected device.
Slow Flash[1]	The port has a failed condition
Fast Flash[2]	A security violation has occurred

1. Slow flash = 0.8 seconds on, 0.8 seconds off
2. Fast flash = 0.4 seconds on, 0.4 seconds off

7.6 Status LEDs

The status LEDs are summarized in Table 7.3.

Table 7.3 Status LEDs

LED	State	Meaning
Power	Off	No power to the switch.
	On	The switch is powered.
Fault	Off	No faults
	On	The Fault LED illuminates briefly after the switch is powered on or reset, at the beginning of self test. If this LED is on for a prolonged time, the switch has encountered a fatal hardware failure, or has failed its self test.
	Fast Flash[1]	A fault has occurred in the switch, the transceiver module (if installed), or a fan. The LED for the device with the fault flashes simultaneously.
Self Test	Off	The normal operational state (the switch is not undergoing self-test).
	On	The switch self test and initialization are in progress after a power cycle or device reset. The switch is not operational until this LED turns off. The Self Test LED also illuminates briefly when a module is “hot swapped” into the device (the module is self tested when it is hot swapped).
	Slow Flash[1]	A component of the switch has failed its self-test. The status LED for that component (a fan, for example), and the switch Fault LED flash simultaneously.
Fan Status	Off	Should never be off. If this LED is off, it indicates a failed LED or circuit.
	On	The fan is operating correctly
	Slow Flash[1]	Indicates a fan failure condition
Security	Off	No security violation
	On	A security violation has occurred
Transceiver	Off	The inserted transceiver is not valid or is not completely seated.
	On	The inserted transceiver module is valid for the product, as indicated by the transceiver ID, and is completely seated.

1. Slow Flash = 0.8 seconds on, 0.8 seconds off. Fast Flash = 0.4 seconds on, 0.4 seconds off.

This Status LEDs are described in more detail in the following paragraphs.

Power

The Power LED is actually two green LEDs directly connected from the 2.5 V supply to ground.

Fault

Software controls the Fault LED from bit 0 (FLED) of the LED_DATA1 register (See Chapter 8, Registers). The LED control multiplexer for the Fault LED does not have a hardware function associated with the fourth input. However, the output of the multiplexer is logically OR'ed with the output of the Fan Hardware Control Multiplexer (See Figure 7.5).

Self Test

Software controls the Self-Test LED from bit 0 (ST) of the LED_DATA2 register (See Chapter 8, Registers). At power-up and reset, this LED is on and remains illuminated until the software and hardware self tests complete successfully. This LED is not driven from the hardware.

Fan

Each fan (if used) has an open drain output that remains LOW if a locked rotor condition exists. This fan output signal is connected to the FAN_FAULTn dedicated input on the L64324. This input can also be sourced by a temperature sensor for a more accurate indication of trouble. When FAN_FAULTn is asserted, the Fan LED and the Fault LED flash in unison at the rate programmed into timer 0 or timer 1, depending on the programming of the Fan hardware control multiplexer (See Figure 7.5). Software controls the Fan hardware control multiplexer from the FAN[1:0] bits of the LED_CNTRL_1B register (See Chapter 8, Registers). The output of the Fan hardware control multiplexer is OR'ed with the output of the Fault LED hardware control multiplexer to create the Fault LED signal for the LED drive logic (See Figure 7.5). The Fan LED is normally always illuminated. When a fault occurs, the Fan LED flashes in unison with the Fault LED.

Security

Software controls the security LED. It is not controlled from hardware. When the CPU detects a security violation, the CPU may illuminate this LED from the SEC bit of the LED_DATA1 register, or software can drive the LED from timer 0 or timer 1.

Transceiver

The transceiver LED illuminates only if the inserted transceiver module is valid for the product, as indicated by the transceiver IDs, and is completely seated. The CPU is responsible for reading the transceiver ID bits and then appropriately programming the X1 and X2 bits in the LED_DATA1 register and the corresponding X1[1:0] and X2[1:0] bits in the LED_CNTRL_1B register.

7.7 Hardware LED Control

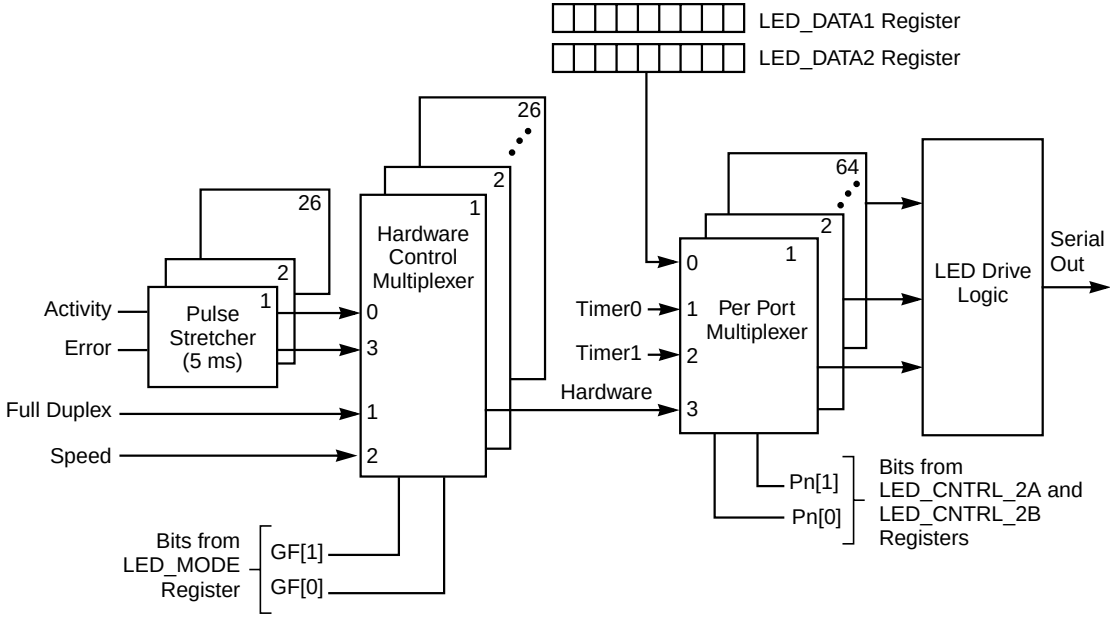
The LED design has been made flexible to allow the LEDs to be driven from a variety of sources, using hardware multiplexers in combination with LED control registers.

7.7.1 Port Mode and Link LEDs Hardware Multiplexer

A per port Hardware Control Multiplexer selects the hardware source that drives the Port Mode LEDs. In addition, 64 additional Per Port multiplexers allow the per port Port Mode and Port Link LEDs to be driven from the selected hardware source, to be flashed based on Timer 0 or Timer 1 intervals, or to be driven from the LED_DATA1 and LED_DATA2 software registers. The selected output bit from each of the 64 multiplexers is then sent to the LED Drive Logic, which sends the data out serially. It is then converted from serial to parallel to drive the appropriate LEDs.

The multiplexers, along with the LED Drive Logic, are shown in Figure 7.2.

Figure 7.2 Mode and Link LED Control



The Hardware Control Multiplexer selects one of four inputs based on the GF[1:0] bits in the LED_MODE register. Software sets the GF[1:0] bits to select Activity, Error, Full Duplex, or Speed events according to the depressions of the Mode switch. Each port has such a multiplexer. The inputs are selected as shown in Table 7.4.

Table 7.4 Multiplexer Operation

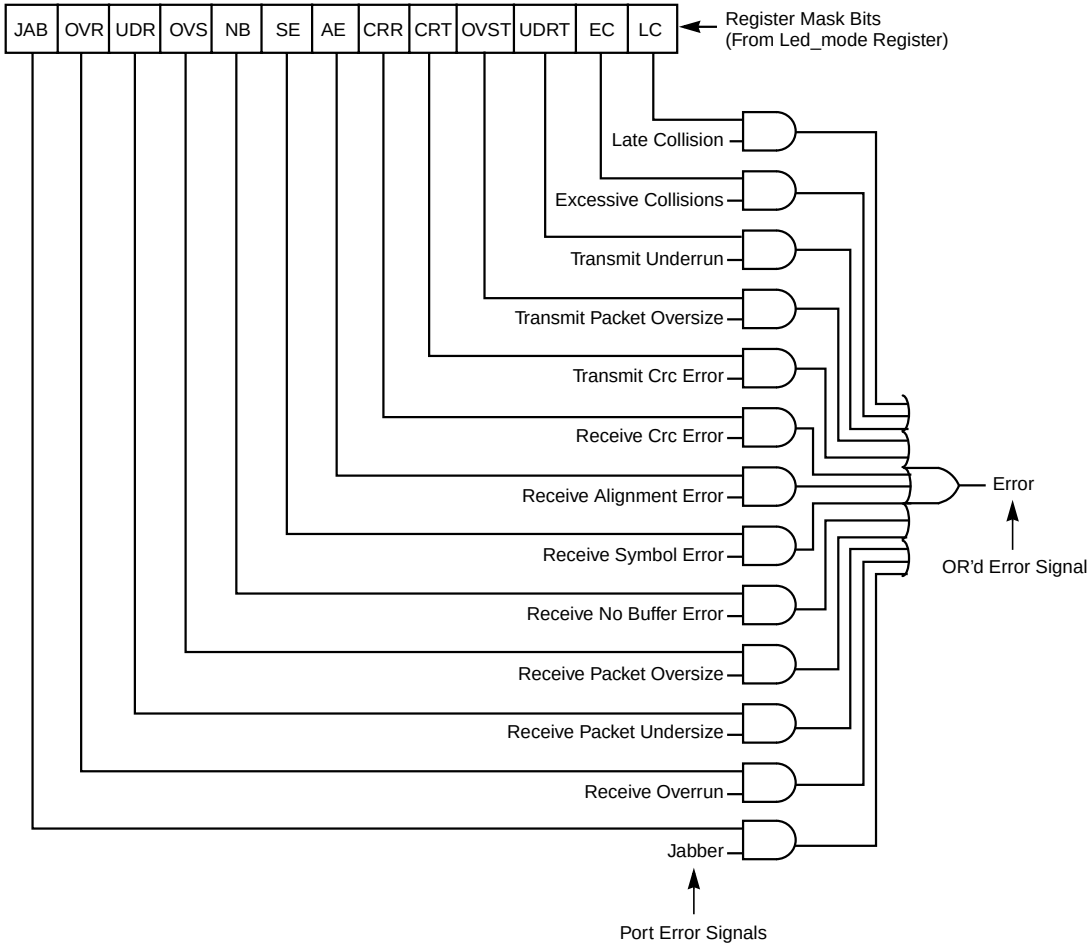
GF[1:0]	Input Selected
0b00	Activity
0b01	Full Duplex
0b10	Speed
0b11	Error

Note that the Activity and Error events are sent through a pulse stretcher, which generates a fixed 5 ms pulse for an Activity or Error event. The 5 ms is a fixed time interval; any activity or error event occurring during the 5 ms does not lengthen the interval past 5 ms. Immediately after the

5 ms duration, the pulse ends. The pulse is then asserted for 5 ms for the next event or if the activity and error events continue beyond the 5 ms. There is no specified off time.

The Error input to the 5 ms pulse stretcher is generated from port logic that monitors a total of 12 signals and uses the LED_MODE register bits to either mask out or enable each signal. If a signal is enabled (not masked) and it is also asserted, the port logic ORs the signal along with all similar signals and asserts the Error input to the pulse stretcher, as shown in Table 7.3. See Chapter 8, Registers for detailed LED_MODE register bit descriptions.

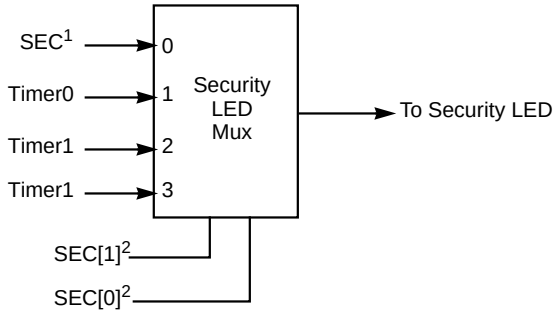
Figure 7.3 Per Port Logic for Error Input to Hardware Control Multiplexer



7.7.2 Security LED Hardware Multiplexer

The Security LED is driven from a multiplexer that can source the LED from the SEC bit of the LED_DATA1 register, or from either Timer 0 or Timer1, as shown in Figure 7.4. The multiplexer is controlled from the SEC[1:0] bits of the LED_CNTRL_1B register.

Figure 7.4 Security LED Control



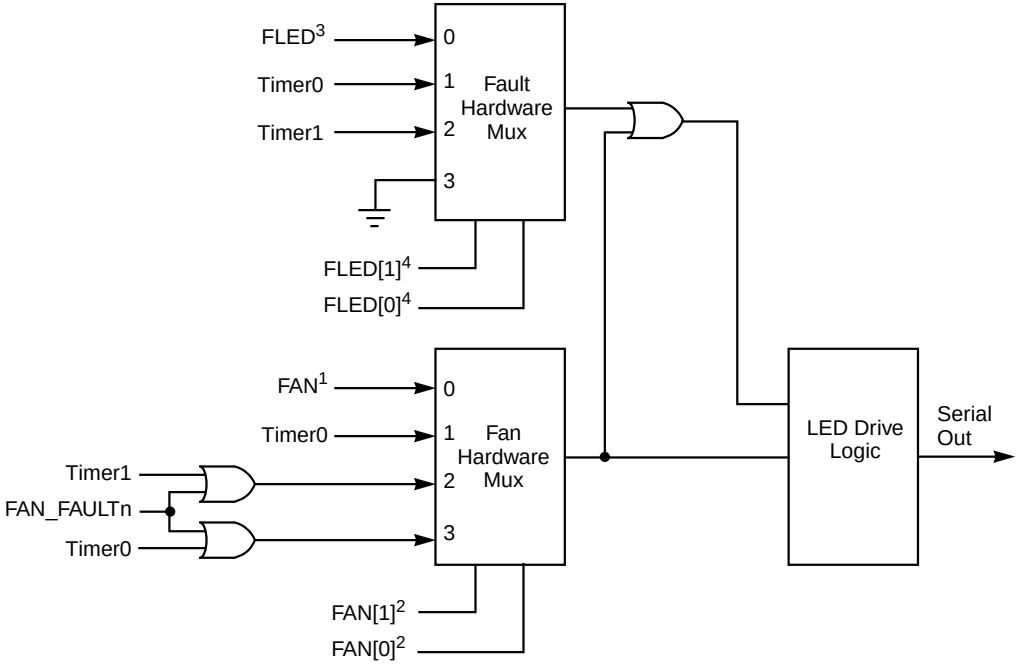
1. The SEC bit is bit 28 of the LED_DATA1 register
2. The SEC[1:0] bits are bits 24 and 25 of the LED_CNTRL_1B register

Note that this is actually implemented with a 3-to-1 multiplexer.

7.7.3 Fan LED

It is important that the Fan LED reflect a fan fault. However, if the temperature of the product has exceeded operational limits as a result of a fan failure, the CPU may not be able to appropriately program the registers to reflect the failure. For this reason, the FAN_FAULTn input is combined with the Timer 0 and Timer1 signals and fed directly to the Fan Hardware Multiplexer, as shown in Figure 7.5. This allows a fan condition to be indicated regardless of the state of the CPU.

Figure 7.5 Fan and Fault LED Control



1. FAN is bit 27 of the LED_DATA1 register
2. The FAN[1:0] bits are bits 22 and 23 of the LED_CNTRL_1B register
3. FLED is bit 0 of the LED_DATA1 register
4. The FLED[1:0] bits are bits 0 and 1 of the LED_CNTRL_1A register

At power-up or reset the FAN bit (bit 27) of the LED_DATA1 register is the default source for driving the Fan LED. It is the responsibility of the CPU to program the Fan Hardware Multiplexer to the 0b10 or 0b11 selection state to allow the FAN_FAULTn signal, combined with the timers, to provide hardware control for the LED. If the CPU is unable to program the LED control registers, all LEDs remain on. This circuit still provides the opportunity for the CPU to control the Fan LED directly or through Timer 0. The output of the Fan Hardware Multiplexer also directly affects the Fault LED, as shown in Figure 7.5.

The CPU can read the state of the FAN_FAULTn signal input through the GPIO2 Status register.

The Fan LED is illuminated when the fan is operating correctly. The Fan LED slow flashes along with the global Fault LED when the FAN_FAULTn input is asserted. The Fan LED is NEVER off.

The FAN_FAULTn input is gated with bit 5 of GPIO2. Because the GPIO pins are completely programmable, so is the FAN_FAULTn input. Specifically, the FAN_FAULTn input may be programmed as active LOW or active HIGH. Programming the polarity of the FAN_FAULTn GPIO pin also alters the operation of the LED circuit that samples the FAN_FAULTn signal. When FAN_FAULTn is programmed as active HIGH, the Fan LED drive logic flashes the Fan LED when the FAN_FAULTn input is asserted HIGH.

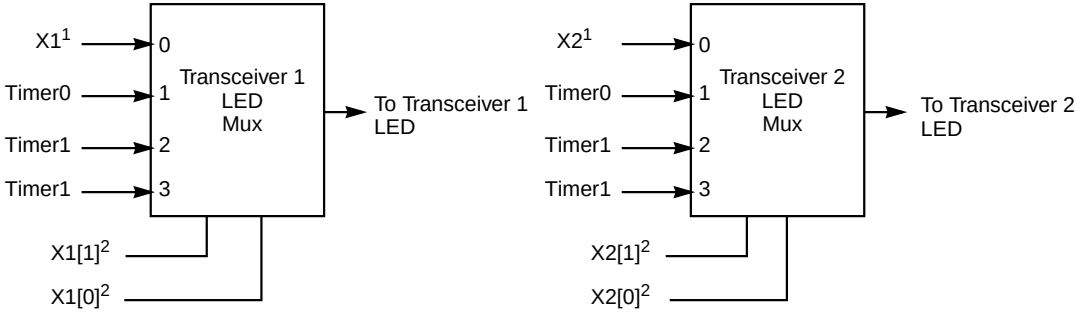
7.7.4 Fault LED

The Fault LED is driven from 0 (FLED) of the LED_DATA1 register, or from Timer 0 or Timer1, as shown in Figure 7.5. The Fault LED Hardware Multiplexer does not have a hardware function associated with the fourth input. However, the output of the multiplexer is logically OR'ed with the output of the Fan Hardware Control Multiplexer and then sent to the LED Drive Logic.

7.7.5 Transceiver LED Hardware Multiplexer

There are two transceiver LEDs. They are each driven from separate multiplexers that can source the LED from the X1 (or X2) bits of the LED_DATA1 register, or from either Timer0 or Timer1, as shown in Figure 7.6. Note that Figure 7.6 shows only one of the two multiplexers. Each multiplexer is controlled from the X1[1:0] (or X2[1:0]) bits of the LED_CNTRL_1B register.

Figure 7.6 Transceiver LED Control



1. The X1 bit is bit 29 of the LED_DATA1 register. X2 is bit 30.
2. The X1[1:0] bits are bits 26 and 27 of the LED_CNTRL_1B register. X2[1:0] are at bit locations 28 and 29.

Note that these are actually 3-to-1 multiplexers

7.8 Port Disable

When a port is disabled, the MAC is disabled and the PHY is isolated. Because the MAC is disabled, packets are not queued to the port and packets are not transmitted. Because the PHY is isolated, no packets are received. Therefore, the Activity and Fault LEDs will not illuminate. While a port is disabled, the Link LED should be off.

While a port is disabled, the Speed and FDX LEDs reflect any forced configuration or the default 10 Mbits/s half-duplex state (LEDs off) if the port is in AutoNegotiation.

7.9 LED Drive Logic

The LED drive logic takes all 64 active HIGH outputs from all of the multiplexers and formats them into a serial data stream. The serial data stream is then sent to an external serial to parallel shift register. The parallel outputs of the shift register directly drive individual LEDs. The data shifted into the shift register is active LOW. Table 7.5 shows the L64324 signals that are used to shift data serially out and load it into an external serial to parallel shift register.

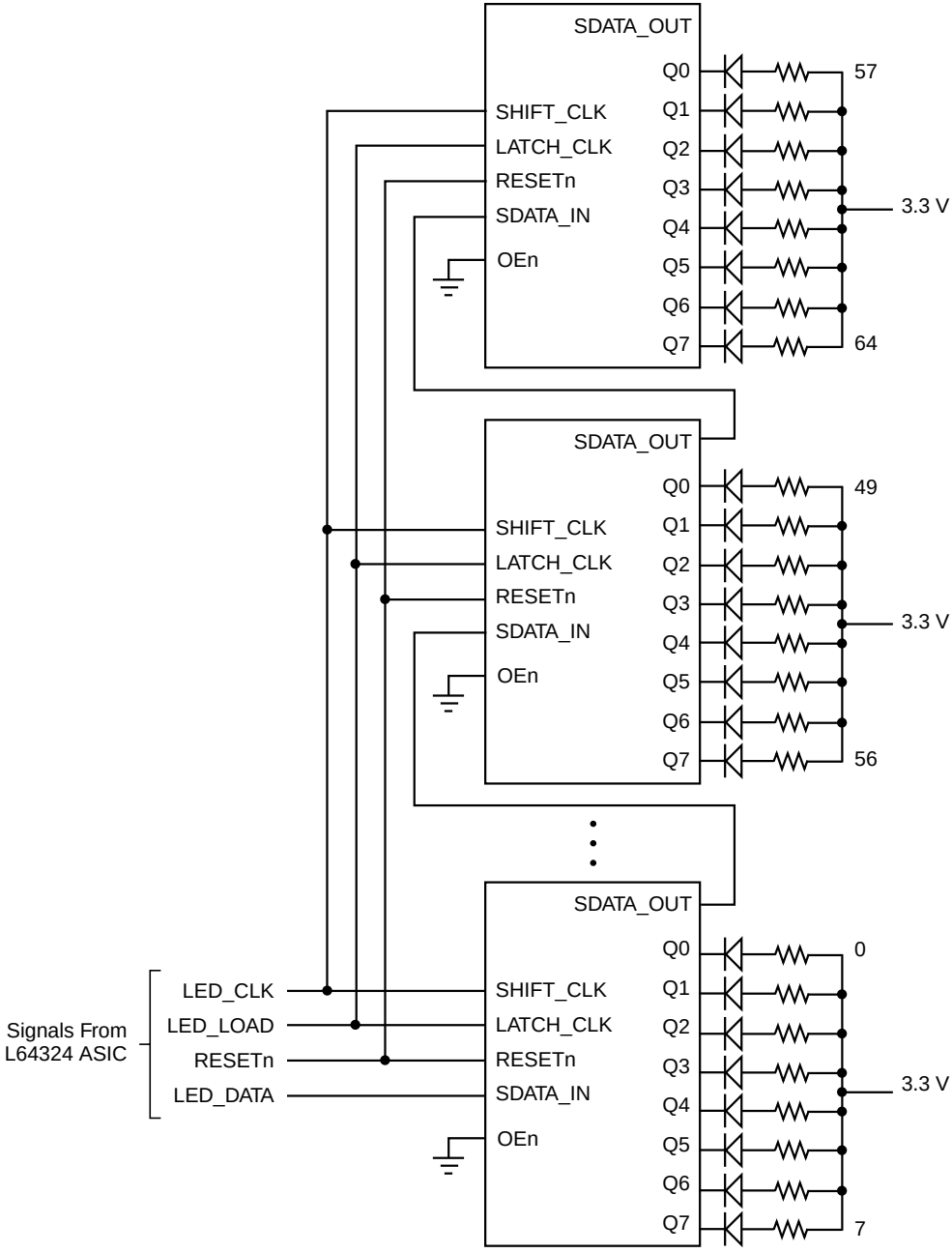
Table 7.5 LED Drive Signals

Signal	Type	Description
LED_CLK	O	This is a 122.07 kHz, 50% duty cycle (approximately) free running clock output that is derived from the 125 MHz system clock.
LED_LOAD	O, L	When serial to parallel devices contain a second stage of latched outputs, this signal is used to load the outputs with the serially chained flip flops. This signal is asserted at the negative edge of LED_CLK, and is asserted for one LED_CLK period. The assertion occurs during the cycle when the 64^{th} data bit is driven on the LED_DATA pin.
LED_DATA	O	This signal contains the 64-bit serial data stream sourced from the LED multiplexers.

The LED drive logic was designed to interface with a 74xx595 serial to parallel shift register. This device contains a serial chain of flip-flops. On each rising edge of the LED_CLK input, the data (LED_DATA) on the SER input is latched into the Q0 flip-flop. The output of the first flip-flop is latched into the D1 input of the second flip-flop, and the Q1 output is latched into D2, and so on. Each Qx output is also input to a secondary output latch. When the serial latches contain the correct LED data, the Qx outputs are latched into the output latches.

Eight 74xx595 devices are required to drive all 64 possible LEDs. Each parallel output of each 74xx595 directly drives a single LED. The active LOW outputs are directly connected to the cathodes of the LEDs. Resistor packs should be used to connect the anode of each LED to 2.5 V. See Figure 7.7.

Figure 7.7 Typical Drive Circuit



This circuit uses a 74xx595 serial to parallel shift register with a secondary latched parallel output. The xx implies any one of a number of families can be used such as LV, F, HC, and so on.

To reduce power, it may be beneficial to connect the LED_CLK signal to the OEn (Output Enable) pin on the 74xx595. This will have the effect of sourcing current to the LEDs only half the time.

Note that the serial data stream can be increased beyond 64 data bits. The extra data bits can be used in place of control signals that require dedicated L64324 pins. Extra bits could also be used for signaling debug information. If extra data is added to the stream, it is important to add the data in 8-bit increments at the beginning of the data stream.

7.10 Timing

Because the LED drive circuitry operates slowly, the timing requirements for it are very relaxed. Figure 7.8 and Table 7.6 shows the timing requirements.

Figure 7.8 LED Timing

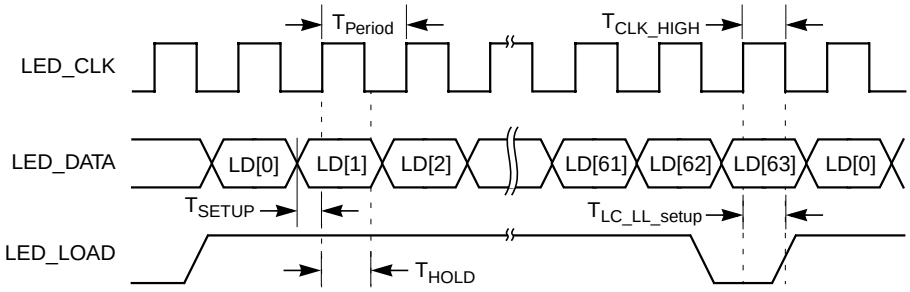


Table 7.6 LED Timing

Timing Parameter	Symbol	Min	Max
LED_DATA to LED_CLK	T_{SETUP}	75 ns	–
LED_DATA hold after LED_CLK	T_{HOLD}	5 ns	–
LED_CLK setup to LED_LOAD	$T_{LC\_LL\_SETUP}$	75 ns	–
LED_CLK (Use 125 MHz/1024: 8.12 µs)	T_{PERIOD}	1000 ns	15.6 µs

As long as the L64324 adheres to the timing above, a 74xx595 from any logic family can be used. Driving the LED_DATA output from the negative edge of LED_CLK easily satisfies the required setup and hold times.

7.11 LED Mapping

The LED Mapping is a function of the final panel mechanical layout of the LEDs. Until a final mechanical layout is complete, the LED mapping shown in Table 7.7 and Table 7.8 must necessarily remain tentative. The tentative mapping in Table 7.8 is based upon the physical port mapping shown in Table 7.7 and the pinout of the 74xx595.

Table 7.7 Mapping of Port Number to Physical Panel Front

1	2	3	4	5	6	…	13	14	15	16	17	18
7	8	9	10	11	12	…	19	20	21	22	23	24

Table 7.8 LED Mapping (Tentative)

Serial Bit	LED	Serial Bit	LED	Serial Bit	LED	Serial Bit	LED
0 (first out)	Link 18	16	Link 14	32	Link 4	48	Link 26
1	Mode 18	17	Mode 14	33	Mode 4	49	Mode 26
2	Link 24	18	Link 20	34	Link 10	50	Xceiver 2
3	Mode 24	19	Mode 20	35	Mode 10	51	Link 25
4	Link 17	20	Link 13	36	Link 3	52	Mode 25
5	Mode 17	21	Mode 13	37	Mode 3	53	Xceiver 1
6	Link 23	22	Link 19	38	Link 9	54	MDIX
7	Mode 23	23	Mode 19	39	Mode 9	55	Error
8	Link 16	24	Link 6	40	Link 2	56	SPD
9	Mode 16	25	Mode 6	41	Mode 2	57	FDX
10	Link 22	26	Link 12	42	Link 8	58	ACT
11	Mode 22	27	Mode 12	43	Mode 8	59	Self-test
12	Link 15	28	Link 5	44	Link 1	60	Fan
13	Mode 15	29	Mode 5	45	Mode 1	61	Security
14	Link 21	30	Link 11	46	Link 7	62	Fault
15	Mode 21	31	Mode 11	47	Mode 7	63 (last out)	Fault

7.12 LED TIMERS

There are two LED timers that provide two hardware-controlled flash rates. Each LED can be individually configured to accept either timer as its source. The configuration is performed through the four LED control registers (LED_CNTRL_1A, LED_CNTRL_1B, LED_CNTRL_2A, and LED_CNTRL_2B). The output of each timer is a square wave with a period that is twice the value programmed into the LED timer fields of the LED_Timers register. See Chapter 8, Registers for more details on the programming of the timer registers.

Chapter 8
Registers

This chapter provides a description of the L64324 24 x 2 Fast Ethernet Intelligent Switch Registers. Table 8.1 is a summary of the registers providing register offset address, register name, CPU read/write, default values and page reference for detailed description.

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3420.0000	RESERVED	R/W	0x0000.0000	8-23
0x3420.0004	RESERVED	R/W	0x0000.0000	8-23
0x3420.0008	RESERVED	R/W	0x0000.0000	8-23
0x3420.000C	MAC_SELECT	R/W	0x0000.0300	8-24
0x3420.0010	GENERAL_PURPOSE	R/W	0x0000.0002	8-26
0x3420.0014	GLOBAL_TIMER	R/W	0x000.0271	8-27
0x3420.0018	PORT_STP_LO	R/W	0x0000.0000	8-28
0x3420.001C	PORT_STP_HI	R/W	0x0000.0000	8-34
0x3420.0020	IGMP_ENABLE	R/W	0x0000.0000	8-37
0x3420.0024	MAC_RESET	R/W	0x0000.0000	8-38
0x3420.0028	RCV_ERR	R/W	0x000.0000	8-39
0x3420.002C	BUS_ERR	R	0x0000.0000	8-40
0x3420.0030	MAC_ADDR1	R/W	0x0000.0xxx	8-41
0x3420.0034	MAC_ADDR2	R/W	0xxxxx.xxxx	8-42
0x3420.0038	LED_DATA1	R/W	0xFFFF.FFFF	8-43
0x3420.003C	LED_DATA2	R/W	0xFFFF.FFFF	8-45

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3420.0040	LED_CNTRL_1A	R/W	0x0000.0000	8-49
0x3420.0044	LED_CNTRL_1B	R/W	0x0000.0000	8-51
0x3420.0048	LED_CNTRL_2A	R/W	0x0000.0000	8-54
0x3420.004C	LED_CNTRL_2B	R/W	0x0000.0000	8-56
0x3420.0050	LED_TIMERS	R/W	0x0026.0013	8-59
0x3420.0054	LED_MODE	W/O	0x0000.11FB	8-60
0x3420.0100	LAN_INT_EN	R/W	0x0000.0000	8-63
0x3420.0104	LAN_INT_ACK	W	0x0000.0000	8-65
0x3420.0108	LAN_INT_STAT	R	0x0000.0000	8-67
0x9000.0000	GPIO1_STAT	R	0x0000.0000	8-70
0x9000.0004	GPIO1_RAW_STAT	R	0x0000.0000	8-71
0x9000.0008	GPIO1_EN_SET	R/W	0x0000.0000	8-72
0x9000.000C	GPIO1_CLEAR	R/W	0x0000.0000	8-73
0x9000.0010	GPIO1_DDR	R/W	0x0000.0000	8-74
0x9000.0014	GPIO1_DATA_OUT	R/W	0x0000.0000	8-75
0x9000.0018	GPIO1_DATA_IN	R	Depends on Input Pull-Up or Pull-Down Resistors on the Board.	8-76
0x9000.001C	GPIO1_POL	R/W	0x0000.0000	8-77
0x9000.0020	GPIO1_EDGE	R/W	0x0000.0000	8-78
0x9000.0024	GPIO1_RESYNC	R/W	0x0000.0000	8-79
0x9000.0040	GPIO2_STAT	R	0x0000.0000	8-80
0x9000.0044	GPIO2_RAW_STAT	R/W	0x0000.0000	8-82
0x9000.0048	GPIO2_EN_SET	R/W	0x0000.0000	8-84
0x9000.004C	GPIO2_CLEAR	W	0x0000.0000	8-85

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x9000.0050	GPIO2_DDR	R/W	0x0000.0000	8-86
0x9000.0054	GPIO2_DATA_OUT	R/W	0x0000.0000	8-89
0x9000.0058	GPIO2_DATA_IN	R	0x0000.0000	8-90
0x9000.005C	GPIO2_POL	R/W	0x0000.0000	8-92
0x9000.0060	GPIO2_EDGE	R/W	0x0000.0000	8-93
0x9000.0064	GPIO2_RESYNC	R/W	0x0000.0000	8-94
0x9000.0080	GPIO3_STAT	R	0x0000.0000	8-95
0x9000.0084	GPIO3_RAW_STAT	R/W	0x0000.0000	8-96
0x9000.0088	GPIO3_EN_SET	R/W	0x0000.0000	8-97
0x9000.008C	GPIO3_CLEAR	W	0x0000.0000	8-98
0x9000.0090	GPIO3_DDR	R/W	0x0000.0000	8-99
0x9000.0094	GPIO3_DATA_OUT	R/W	0x0000.0000	8-100
0x9000.0098	GPIO3_DATA_IN	R	0x0000.0000	8-101
0x9000.009C	GPIO3_POL	R/W	0x0000.0000	8-103
0x9000.00A0	GPIO3_EDGE	R/W	0x0000.0000	8-104
0x9000.00A4	GPIO3_RESYNC	R/W	0x0000.0000	8-105
0x9000.00C0	GPIO4_STAT	R	0x0000.0000	8-106
0x9000.00C4	GPIO4_RAW_STAT	R/W	0x0000.0000	8-108
0x9000.00C8	GPIO4_EN_SET	R/W	0x0000.0000	8-110
0x9000.00CC	GPIO4_CLEAR	W	0x0000.0000	8-111
0x9000.00D0	GPIO4_DDR	R/W	0x0000.0000	8-112
0x9000.00D4	GPIO4_DATA_OUT	R/W	0x0000.00x0	8-113
0x9000.00D8	GPIO4_DATA_IN	R	0x0000.0000	8-115
0x9000.00DC	GPIO4_POL	R	0x0000.0000	8-117
0x9000.00E0	GPIO4_EDGE	R/W	0x0000 00FF	8-118

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x9000.00E4	GPIO4_RESYNC	R/W	0x0000 00FF	8-119
0x8000.0000/0x8000.0100	IRQ_STAT/FIQ_STAT	R	0x0xxx.xxxx	8-122
0x8000.0004/0x8000.0104	IRQ_RAW_STAT/FIQ_RAW_STAT	R/W	0x0x0.0000	8-124
0x8000.008/0x8000.0108	IRQ_EN_SET/FIQ_EN_SET	R/W	0x0000.0000	8-125
0x8000.0010	FIQ_IRQ_SWI	R/W	0x0000.0000	8-126
0x8000.000C/0x8000.010C	IRQ_CLEAR/FIQ_CLEAR	R/W	0x0000.0000	8-127
0x8400.0000	TIMER_1	R/W	0x0000.xxxx	8-129
0x8400.0004	TIMER_1_PSCALE	R/W	0x0000.xxxx	8-130
0x8400.0008	TIMER_1_CNTL	R/W	0x0000.000xx	8-131
0x8400.000C	TIMER_2	R/W	0x0000.00xx	8-133
0x8400.0010	TIMER_2_PSCALE	R/W	0x0000.00xx	8-134
0x8400.0014	TIMER_2_CNTL	R/W	0x0000.000x	8-135
0x8400.0018	TIMER_TRG_1	R/W	0x0000.xxxx	8-137
0x8400.001C	TIMER_TRG_2	R/W	0x0000.xxxx	8-138
0x8400.0020	TIMER_TRG_3	R/W	0x0000.xxxx	8-139
0x8400.0024	TIMER_TRG_4	R/W	0x0000.xxxx	8-140
0x8400.0028	TIMER_TRG_5	R/W	0x0000.xxxx	8-141
0x8400.002C	TIMER_TRG_6	R/W	0x0000.xxxx	8-142
0x8400.0030	TIMER_TRG_CNTL	R/W	0x0000.00xx	8-143
0x8800.0000	REMAP_REG	R/W	0xxxxx.xxxx	8-146
0x8800.0010	REMAP_REV	R	0x0000 0001	8-147
0x8800.0020	REMAP_REMAP	W	0xxxxx.xxxx	8-148
0x8800.0030	REMAP_RST_STAT	R/W	0x0000.0001	8-149

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x8800.0034	REMAP_RST_CLR	R/W	0x0000.0000	8-151
0x8800.0038	REMAP_SRSR	R/W	0x0000.0000	8-152
0x8800.003C	REMAP_SRCR	W	0x0000.0000	8-153
0x8800.0040	REMAP_HCR	R/W	0x0000.000x	8-154
0x8C00.0000	UART_RHR (Receive Holding Register)	R	0xxxxx.xxxx	8-156
0x8C00.0000	UART_THR (Transmit Holding Register)	W	0xxxxx.xxxx	8-157
0x8C00.0004	UART_IER (Interrupt Enable Register)	R/W	0x0000.000x	8-158
0x8C00.0000	UART_LDR (LSB Divisor Latch Register)	W	0xxxxx.xxxx	8-159
0x8C00.0004	UART_MDLR (MSB Divisor Latch Register)	W	0xxxxx.xxxx	8-160
0x8C00.0008	UART_ISR (Interrupt Status Register)	R	0x0000.000x	8-161
0x8C00.0008	UART_FCR (FIFO Control Register)	W	0x0000.00xx	8-162
0x8C00.000C	UART_LCR (Line Control Register)	W	0x0000.00xx	8-165
0x8C00.0010	UART_MCR (Modem Control Register)	W	0x0000.000x	8-167
0x8C00.0014	UART_LSR (Line Status Register)	R	0x0000.00xx	8-169
0x8C00.0018	UART_MSR (Modem Status Register)	R/W	0x0000.00xx	8-171
0x8C00.001C	UART_SRn (Scratch Register)	R/W	0xxxxx.xxxx	8-173
0x8C00.0020	UART_SSR (Speed Sense Register)	R/W	0x0000.xxxx	8-174
0x340X.0000	RMON_TOTAL_COUNT	R/W	0x0000.0000	8-177
0x340X.0004	RMON_LENGTH64	R/W	0x0000.0000	8-177
0x340X.0008	RMON_LENGTH65	R/W	0x0000.0000	8-177
0x340X.000C	RMON_LENGTH128	R/W	0x0000.0000	8-178
0x340X.0010	RMON_LENGTH256	R/W	0x0000.0000	8-178
0x340X.0014	RMON_LENGTH512	R/W	0x0000.0000	8-178
0x340X.0018	RMON_LENGTH1024	R/W	0x0000.0000	8-179
0x340X.001C	RMON_LENGTH1519	R/W	0x0000.0000	8-179

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x340X.0020	RMON_SYMBOL_ERROR	R/W	0x0000.0000	8-179
0x340X.0024	RMON_CRC_ERROR	R/W	0x0000.0000	8-180
0x340X.0028	RMON_UNDERSIZE	R/W	0x0000.0000	8-180
0x340X.002C	RMON_OVERSIZE	R/W	0x0000.0000	8-180
0x340X.0030	RMON_JABBER	R/W	0x0000.0000	8-181
0x340X.0034	RMON_FRAGMENT	R/W	0x0000.0000	8-181
0x340X.0038	RMON_RX_COUNT	R/W	0x0000.0000	8-181
0x340X.003C	RMON_RX_BRDCAST	R/W	0x0000.0000	8-182
0x340X.0040	RMON_RX_MLTCAST	R/W	0x0000.0000	8-182
0x340X.0044	RMON_RX_UNICAST	R/W	0x0000.0000	8-182
0x340X.0048	RMON_RX_PAUSE	R/W	0x0000.0000	8-183
0x340X.004C	RMON_RX_UNKNOWN	R/W	0x0000.0000	8-183
0x340X.0050	RMON_RX_OVERFLOW	R/W	0x0000.0000	8-183
0x340X.0054	RMON_FALSE_CARRIER	R/W	0x0000.0000	8-184
0x340X.0058	RMON_RX_PKT_DROP	R/W	0x0000.0000	8-184
0x340X.005C	RMON_TX_COUNT	R/W	0x0000.0000	8-184
0x340X.0060	RMON_TX_UNDER_RUN	R/W	0x0000.0000	8-185
0x340X.0064	RMON_TX_BRDCAST	R/W	0x0000.0000	8-185
0x340X.0068	RMON_TX_MLTCAST	R/W	0x0000.0000	8-185
0x340X.006C	RMON_TX_UNICAST	R/W	0x0000.0000	8-186
0x340X.0070	RMON_TX_PAUSE	R/W	0x0000.0000	8-186
0x340X.0074	RMON_DRIBBLE_ERR	R/W	0x0000.0000	8-186
0x340X.0078	RMON_TX_DEFER	R/W	0x0000.0000	8-187
0x340X.007C	RMON_TX_LATE_COL	R/W	0x0000.0000	8-187
0x340X.0080	RMON_SINGLE_COL	R/W	0x0000.0000	8-187

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x340X.0084	RMON_MULTI_COL	R/W	0x0000.0000	8-188
0x340X.0088	RMON_XSIVE_COL	R/W	0x0000.0000	8-188
0x340X.008C	RMON_TOTAL_COLLISION	R/W	0x0000.0000	8-188
0x340X.0800	E110_SMII_STATUS	R	0x0000.000x	8-189
0x340X.0810	PVID_PPD (E110 Port Default VLAN and Priority)	R/W	0x000.00xx	8-190
0x340X.0820	E110_MAC_CFG_1	R/W	0x0004.888B	8-191
0x340X.0830	E110_MAC_CFG_2	R/W	0x00xx.xx00	8-192
0x340X.1000	E110_TX_FIFO	R/W	0x0000.0000	8-194
0x340X.1800	E110_RX_FIFO	R/W	0x0000.0000	8-194
0x341X.0000	RMON_TOTAL_COUNT	R/W	0x0000.0000	8-196
0x341X.0004	RMON_LENGTH64	R/W	0x0000.0000	8-196
0x341X.0008	RMON_LENGTH65	R/W	0x0000.0000	8-196
0x341X.000C	RMON_LENGTH128	R/W	0x0000.0000	8-197
0x341X.0010	RMON_LENGTH256	R/W	0x0000.0000	8-197
0x341X.0014	RMON_LENGTH512	R/W	0x0000.0000	8-197
0x341X.0018	RMON_LENGTH1024	R/W	0x0000.0000	8-198
0x341X.001C	RMON_LENGTH1519	R/W	0x0000.0000	8-198
0x341X.0020	RMON_SYMBOL_ERROR	R/W	0x0000.0000	8-198
0x341X.0024	RMON_CRC_ERROR	R/W	0x0000.0000	8-199
0x341X.0028	RMON_UNDERSIZE	R/W	0x0000.0000	8-199
0x341X.002C	RMON_OVERSIZE	R/W	0x0000.0000	8-199
0x341X.0030	RMON_JABBER	R/W	0x0000.0000	8-200
0x341X.0034	RMON_FRAGMENT	R/W	0x0000.0000	8-200
0x341X.0038	RMON_RX_COUNT	R/W	0x0000.0000	8-200
0x341X.003C	RMON_RX_BRDCAST	R/W	0x0000.0000	8-201

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x341X.0040	RMON_RX_MLTCAST	R/W	0x0000.0000	8-201
0x341X.0044	RMON_RX_UNICAST	R/W	0x0000.0000	8-201
0x341X.0048	RMON_RX_PAUSE	R/W	0x0000.0000	8-202
0x341X.004C	RMON_RX_UNKNOWN	R/W	0x0000.0000	8-202
0x341X.0050	RMON_RX_OVERFLOW	R/W	0x0000.0000	8-202
0x341X.0054	RMON_FALSE_CARRIER	R/W	0x0000.0000	8-203
0x341X.0058	RMON_RX_PKT_DROP	R/W	0x0000.0000	8-203
0x341X.005C	RMON_TX_COUNT	R/W	0x0000.0000	8-203
0x341X.0060	RMON_TX_UNDER_RUN	R/W	0x0000.0000	8-204
0x341X.0064	RMON_TX_BRDCAST	R/W	0x0000.0000	8-204
0x341X.0068	RMON_TX_MLTCAST	R/W	0x0000.0000	8-204
0x341X.006C	RMON_TX_UNICAST	R/W	0x0000.0000	8-205
0x341X.0070	RMON_TX_PAUSE	R/W	0x0000.0000	8-205
0x341X.0800	GIG_CFG	R/W	0x0000.019C	8-205
0x341X.0804	GIG_PCS_HW_CFG (Gigabit PCS Hardware Link Configuration)	R/W	0x0000.1840	8-207
0x341X.0808	GIG_PCS_SW_CFG (Gigabit PCS Software Link Configuration)	R/W	0x0000.0000	8-209
0x341X.080C	GIG_PCS_SW_TXCFG (Gigabit PCS Transmit Configuration)	R/W	0x0000.0000	8-210
0x341X.0810	GIG_PCS_SW_STAT (Gigabit PCS Software Link Status)	R	0x0000.0000	8-211
0x341X.0814	PVID_PPD (Gigabit Port Default VLAN and Priority)	R/W	0x0000.0001	8-212
0x341X.0818	MAC_ADDR_LO (Gigabit Port MAC Address Bits[4:0])	R/W	Port_ID	8-213
0x341X.1000	GIGA_TX_FIFO (Gigabit Transmit FIFO (base))	R/W	00x0000.0000	8-214

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x341X.1800	GIGA_RX_FIFO (Gigabit Receive FIFO (base))	R/W	0x0000.0000	8-214
0x341X.2000–0x341X.2230	GIGA_EXTRACTOR_REGS (Gigabit Extractor Registers)	R	0x0000.0000	8-215
0x3426.0000	System Flow Control	R/W	0x0000.0000	8-220
0x3426.0004	CBA	R/W	0x0000.0FFF	8-221
0x3426.0008	CPU_BUF_REQ	R/W	0x000.0000	8-222
0x3426.000C	BUF_RET (Buffer Return)	R/W	0x00.0000	8-223
0x3426.4000–0x3426.7FFF	ABIT Memory Location	R/W	0x0000.0000	8-224
0x3427.0000	SDRAM_TIMING_REG (SDRAM Timing)	R/W	0x0006.389B	8-226
0x3427.0004	SDRAM_MODE_REG (SDRAM Mode)	R/W	0x0000.0033	8-227
0x3427.0008	SDRAM_REFRESH_REG (SDRAM Refresh)	R/W	0x0000.07D1	8-229
0x3427.000C	Reserved	R/W	0x0000.0000	8-230
0x3427.0010	RESERVED	R/W	0x0000.0000	8-231
0x3427.0020	FLASH_CONFIG (Flash Configuration)	R/W	Board Configuration Pins	8-232
0x3428.0000	LAST_TX_STATUS (Last Transmit Status)	R	0x3818.2000	8-234
0x3428.0080	ARBITRATOR_FAIRNESS (DMA Arbitrator Fairness)	R/W	0x9A00.0318	8-235
0x3428.0088	PREFETCH_BUFF_EN (DMA Prefetch Buffer/Write Buffer Enable)	R/W	0xC000.0000	8-236
0x3428.0340–0x3428.039F	WRITE_BUFF (DMA Write Buffer)	R/W	Unknown	8-237
0x3428.0640–0x3428.069F	PREFETCH_BUFF (DMA Prefetch Buffer)	R/W	Unknown	8-238
0x3425.0000	ADDRESS (MAC Address)	R/W	0xxxxx.xxxx	8-240
0x3425.0004	ADDR_CMD (Commands and MAC Address)	R/W	0x0xx0.xxxx	8-241
0x3425.0008	UC1/MC1 (ARL Unicast/Multicast Word 1)	R/W[12:0], R[31:11]	0x0xx0.0xxx	8-243

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3425.000C	UC 2/MC 2 (ARL Unicast/Multicast Word 2)	R/W	0xxxxx.0xxx	8-245
0x3425.0010	UC 3/MC 3 (ARL Unicast/Multicast Word 3)	R/W	0xxxxx.xxxx	8-246
0x3425.0014	FIELD_SEL (Field Selector)	R/W	0x0000.0000	8-247
0x3425.0018	PKT_TX_1 (Transmit Descriptor First Word)	R/W	0xx0xx.xxxx	8-249
0x3425.001C	PKT_TX_2 (Transmit Descriptor Second Word)	R/W	0x00xx.xxx0	8-251
0x3425.0020	PKT_TX 3 (Transmit Descriptor Third Word)	R/W	0x0000.xxxx	8-252
0x3425.0024	PKT_TX_4 (Transmit Descriptor Fourth Word)	R/W	0xxxxx.xxxx	8-254
0x3425.0028	PKT_TX_5 (Transmit Descriptor Fifth Word)	R/W	0xxxxx.xxxx	8-255
0x3425.002C	PKT_TX_6 (Transmit Descriptor Sixth Word)	R/W	0xxxxx.xxxx	8-256
0x3425.0030	ERR_EN_MASK (Error Enable Mask)	R/W	0x0000.0000	8-257
0x3425.0034	SRC_PORT_LOCK (Source Port Locking)	R/W	0x0000.0000	8-259
0x3425.0038	PORT_AGE_CTRL (Port Aging Control)	R/W	0x0000.0000	8-260
0x3425.003C	AGE_INTERVAL (Aging Interval)	R/W	0xx6xx.003C	8-261
0x3425.0040	AGE_TIMER (Aging Interval Timer)	Read Only	0xxxx0.0000	8-262
0x3425.0044	PORT_MIRR_MASK (Port Mirror Mask)	R/W	0x0000.0000	8-263
0x3425.0048	LMPR (Logical Map Physical)	R/W	0x0000.0000	8-264
0x3425.004C	VLAN_ING_ENABLE (VLAN Ingress Enable)	R/W	0x0000.0000	8-265
0x3425.0050–0x3425.0057	RESERVED	-	0xxxxx.xxxx	8-266
0x3425.0058	RMON_SKIPCNT_1 (RMON Skip Count Port 1)	R/W	0xxxxx.xx00	8-267
0x3425.005C	RMON_SKIPCNT_2 (RMON Skip Count Port 2)	R/W	0xxxxx.xx00	8-267
0x3425.0060	RMON_SKIPCNT_3 (RMON Skip Count Port 3)	R/W	0xxxxx.xx00	8-268
0x3425.0064	RMON_SKIPCNT_4 (RMON Skip Count Port 4)	R/W	0xxxxx.xx00	8-268

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3425.0068	RMON_SKIPCNT_5 (RMON Skip Count Port 5)	R/W	0xxxxx.xx00	8-268
0x3425.006C	RMON_SKIPCNT_6 (RMON Skip Count Port 6)	R/W	0xxxxx.xx00	8-269
0x3425.0070	RMON_SKIPCNT_7 (RMON Skip Count Port 7)	R/W	0xxxxx.xx00	8-269
0x3425.0074	RMON_SKIPCNT_8 (RMON Skip Count Port 8)	R/W	0xxxxx.xx00	8-269
0x3425.0078	RMON_SKIPCNT_9 (RMON Skip Count Port 9)	R/W	0xxxxx.xx00	8-270
0x3425.007C	RMON_SKIPCNT_10 (RMON Skip Count Port 10)	R/W	0xxxxx.xx00	8-270
0x3425.0080	RMON_SKIPCNT_11 (RMON Skip Count Port 11)	R/W	0xxxxx.xx00	8-270
0x3425.0084	RMON_SKIPCNT_12 (RMON Skip Count Port 12)	R/W	0xxxxx.xx00	8-271
0x3425.0088	RMON_SKIPCNT_13 (RMON Skip Count Port 13)	R/W	0xxxxx.xx00	8-271
0x3425.008C	RMON_SKIPCNT_14 (RMON Skip Count Port 14)	R/W	0xxxxx.xx00	8-271
0x3425.0090	RMON_SKIPCNT_15 (RMON Skip Count Port 15)	R/W	0xxxxx.xx00	8-272
0x3425.0094	RMON_SKIPCNT_16 (RMON Skip Count Port 16)	R/W	0xxxxx.xx00	8-272
0x3425.0098	RMON_SKIPCNT_17 (RMON Skip Count Port 17)	R/W	0xxxxx.xx00	8-272
0x3425.009C	RMON_SKIPCNT_18 (RMON Skip Count Port 18)	R/W	0xxxxx.xx00	8-273
0x3425.00A0	RMON_SKIPCNT_19 (RMON Skip Count Port 19)	R/W	0xxxxx.xx00	8-273
0x3425.00A4	RMON_SKIPCNT_20 (RMON Skip Count Port 20)	R/W	0xxxxx.xx00	8-273

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3425.00A8	RMON_SKIPCNT_21 (RMON Skip Count Port 21)	R/W	0xxxxx.xx00	8-274
0x3425.00AC	RMON_SKIPCNT_22 (RMON Skip Count Port 22)	R/W	0xxxxx.xx00	8-274
0x3425.00B0	RMON_SKIPCNT_23 (RMON Skip Count Port 23)	R/W	0xxxxx.xx00	8-274
0x3425.00B4	RMON_SKIPCNT_24 (RMON Skip Count Port 24)	R/W	0xxxxx.xx00	8-275
0x3425.00B8	RMON_SKIPCNT_25 (RMON Skip Count Port 25)	R/W	0xxxxx.xx00	8-275
0x3425.00BC	RMON_SKIPCNT_26 (RMON Skip Count Port 26)	R/W	0xxxxx.xx00	8-275
0x3425.0190	IF_OUT_DISCARDS_1 (Output Packet Discards, Port 1)	Read Clear/ Read Only	0x0000.0000	8-276
0x3425.0194	IF_OUT_DISCARDS_2 (Output Packet Discards, Port 2)	Read Clear/ Read Only	0x0000.0000	8-276
0x3425.0198	IF_OUT_DISCARDS_3 (Output Packet Discards, Port 3)	Read Clear/ Read Only	0x0000.0000	8-276
0x3425.019C	IF_OUT_DISCARDS_4 (Output Packet Discards, Port 4)	Read Clear/ Read Only	0x0000.0000	8-277
0x3425.01A0	IF_OUT_DISCARDS_5 (Output Packet Discards, Port 5)	Read Clear/ Read Only	0x0000.0000	8-277
0x3425.01A4	IF_OUT_DISCARDS_6 (Output Packet Discards, Port 6)	Read Clear/ Read Only	0x0000.0000	8-277
0x3425.01A8	IF_OUT_DISCARDS_7 (Output Packet Discards, Port 7)	Read Clear/ Read Only	0x0000.0000	8-278
0x3425.01AC	IF_OUT_DISCARDS_8 (Output Packet Discards, Port 8)	Read Clear/ Read Only	0x0000.0000	8-278

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3425.01B0	IF_OUT_DISCARDS_9 (Output Packet Discards, Port 9)	Read Clear/ Read Only	0x0000.0000	8-278
0x3425.01B4	IF_OUT_DISCARDS_10 (Output Packet Discards, Port 10)	Read Clear/ Read Only	0x0000.0000	8-279
0x3425.01B8	IF_OUT_DISCARDS_11 (Output Packet Discards, Port 11)	Read Clear/ Read Only	0x0000.0000	8-279
0x3425.01BC	IF_OUT_DISCARDS_12 (Output Packet Discards, Port 12)	Read Clear/ Read Only	0x0000 0000	8-279
0x3425.01C0	IF_OUT_DISCARDS_13 (Output Packet Discards, Port 13)	Read Clear/ Read Only	0x0000.0000	8-280
0x3425.01C4	IF_OUT_DISCARDS_14 (Output Packet Discards, Port 14)	Read Clear/ Read Only	0x0000.0000	8-280
0x3425.01C8	IF_OUT_DISCARDS_15 (Output Packet Discards, Port 15)	Read Clear/ Read Only	0x0000.0000	8-280
0x3425.01CC	IF_OUT_DISCARDS_16 (Output Packet Discards, Port 16)	Read Clear/ Read Only	0x0000.0000	8-281
0x3425.01D0	IF_OUT_DISCARDS_17 (Output Packet Discards, Port 17)	Read Clear/ Read Only	0x0000.0000	8-281
0x3425.01D4	IF_OUT_DISCARDS_18 (Output Packet Discards, Port 18)	Read Clear/ Read Only	0x0000.0000	8-281
0x3425.01D8	IF_OUT_DISCARDS_19 (Output Packet Discards, Port 19)	Read Clear/ Read Only	0x0000.0000	8-282
0x3425.01DC	IF_OUT_DISCARDS_20 (Output Packet Discards, Port 20)	Read Clear/ Read Only	0x0000.0000	8-282

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3425.01E0	IF_OUT_DISCARDS_21 (Output Packet Discards, Port 21)	Read Clear/ Read Only	0x0000.0000	8-282
0x3425.01E4	IF_OUT_DISCARDS_22 (Output Packet Discards, Port 22)	Read Clear/ Read Only	0x0000.0000	8-283
0x3425.01E8	IF_OUT_DISCARDS_23 (Output Packet Discards, Port 23)	Read Clear/ Read Only	0x0000.0000	8-283
0x3425.01EC	IF_OUT_DISCARDS_24 (Output Packet Discards, Port 24)	Read Clear/ Read Only	0x0000.0000	8-283
0x3425.01F0	IF_OUT_DISCARDS_25 (Output Packet Discards, Port 25)	Read Clear/ Read Only	0x0000.0000	8-284
0x3425.01F4	IF_OUT_DISCARDS_26 (Output Packet Discards, Port 26)	Read Clear/ Read Only	0x0000.0000	8-284
0x3425.01F8	PORT_IN_DISCARDS_1 (Input Packet Port Discards, Port 1)	Read Clear/ Read Only	0x0000.0000	8-285
0x3425.01FC	PORT_IN_DISCARDS_2 (Input Packet Port Discards, Port 2)	Read Clear/ Read Only	0x0000.0000	8-285
0x3425.0200	PORT_IN_DISCARDS_3 (Input Packet Port Discards, Port 3)	Read Clear/ Read Only	0x0000.0000	8-286
0x3425.0204	PORT_IN_DISCARDS_4 (Input Packet Port Discards, Port 4)	Read Clear/ Read Only	0x0000.0000	8-286
0x3425.0208	PORT_IN_DISCARDS_5 (Input Packet Port Discards, Port 5)	Read Clear/ Read Only	0x0000.0000	8-286
0x3425.020C	PORT_IN_DISCARDS_6 (Input Packet Port Discards, Port 6)	Read Clear/ Read Only	0x0000.0000	8-287

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3425.0210	PORT_IN_DISCARDS_7 (Input Packet Port Discards, Port 7)	Read Clear/ Read Only	0x0000.0000	8-287
0x3425.0214	PORT_IN_DISCARDS_8 (Input Packet Port Discards, Port 8)	Read Clear/ Read Only	0x0000.0000	8-287
0x3425.0218	PORT_IN_DISCARDS_9 (Input Packet Port Discards, Port 9)	Read Clear/ Read Only	0x0000.0000	8-288
0x3425.021C	PORT_IN_DISCARDS_10 (Input Packet Port Discards, Port 10)	Read Clear/ Read Only	0x0000.0000	8-288
0x3425.0220	PORT_IN_DISCARDS_11 (Input Packet Port Discards, Port 11)	Read Clear/ Read Only	0x0000.0000	8-288
0x3425.0224	PORT_IN_DISCARDS_12 (Input Packet Port Discards, Port 12)	Read Clear/ Read Only	0x0000.0000	8-289
0x3425.0228	PORT_IN_DISCARDS_13 (Input Packet Port Discards, Port 13)	Read Clear/ Read Only	0x0000.0000	8-289
0x3425.022C	PORT_IN_DISCARDS_14 (Input Packet Port Discards, Port 14)	Read Clear/ Read Only	0x0000.0000	8-289
0x3425.0230	PORT_IN_DISCARDS_15 (Input Packet Port Discards, Port 15)	Read Clear/ Read Only	0x0000.0000	8-290
0x3425.0234	PORT_IN_DISCARDS_16 (Input Packet Port Discards, Port 16)	Read Clear/ Read Only	0x0000.0000	8-290
0x3425.0238	PORT_IN_DISCARDS_17 (Input Packet Port Discards, Port 17)	Read Clear/ Read Only	0x0000.0000	8-290
0x3425.023C	PORT_IN_DISCARDS_18 (Input Packet Port Discards, Port 18)	Read Clear/ Read Only	0x0000.0000	8-291

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3425.0240	PORT_IN_DISCARDS_19 (Input Packet Port Discards, Port 19)	Read Clear/ Read Only	0x0000.0000	8-291
00x3425.0244	PORT_IN_DISCARDS_20 (Input Packet Port Discards, Port 20)	Read Clear/ Read Only	0x0000.0000	8-291
0x3425.0248	PORT_IN_DISCARDS_21 (Input Packet Port Discards, Port 21)	Read Clear/ Read Only	0x0000.0000	8-292
0x3425.024C	PORT_IN_DISCARDS_22 (Input Packet Port Discards, Port 22)	Read Clear/ Read Only	0x0000.0000	8-292
0x3425.0250	PORT_IN_DISCARDS_23 (Input Packet Port Discards, Port 23)	Read Clear/ Read Only	0x0000.0000	8-292
0x3425.0254	PORT_IN_DISCARDS_24 (Input Packet Port Discards, Port 24)	Read Clear/ Read Only	0x0000.0000	8-293
0x3425.0258	PORT_IN_DISCARDS_25 (Input Packet Port Discards, Port 25)	Read Clear/ Read Only	0x0000.0000	8-293
0x3425.025C	PORT_IN_DISCARDS_26 (Input Packet Port Discards, Port 26)	Read Clear/ Read Only	0x0000.0000	8-293
0x3425.0260	LEARNED_ENTRY_DISCARDS	Read Clear/ Read Only	0x0000.0000	8-294
0x3425.0300–0x3425.03FF	VLAN_TABLE_ENTRY (VLAN Table Entry Registers)	R/W	0xxxxx.xxxx	8-295
0x3425.0400–0x3425.047F	VLAN_CAM (VLAN CAM Registers)	R/W	0xxxxx.xxxx	8-299
0x3425.8000–0x3425.FFFF	ARL_TABLE	R/W	0xxxxx.xxxx	8-300
0x3424.0000	RX_EN (Receive Queue Enables)	R/W	0x0000.000F	8-307
0x3424.0004	RX1_BASE (Base Index for Ring 1)	R/W	0x0000.0040	8-309

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3424.0008	RX2_BASE (Base Index for Ring 2)	R/W	0x0000.0080	8-310
0x3424.000C	RX3_BASE (Base Index for Ring 3)	R/W	0x0000.00C0	8-311
0x3424.0010	RX0_FILL	R/W	0x0000.0001	8-312
0x3424.0014	RX1_FILL	R/W	0x0000.0041	8-313
0x3424.0018	RX2_FILL	R/W	0x0000.0081	8-314
0x3424.001C	RX3_FILL	R/W	0x0000.00C1	8-315
0x3424.0020	RX0_DRAIN	R/W	0x0000.0000	8-316
0x3424.0024	RX1_DRAIN	R/W	0x0000.0040	8-317
0x3424.0028	RX2_DRAIN	R/W	0x0000.0080	8-318
0x3424.002C	RX3_DRAIN	R/W	0x0000.00C0	8-319
0x3424.0300	TDAC (Transmit Descriptor Aging Count Register)	R/W	0x0000.0004	8-320
0x3424.0304	DDRPR (Desired Descriptor Ring Pointer Register)	R/W	0x0000.7000	8-321
0x3424.0308	SPR (Starvation Prevention Register)	R/W	0x0000.0404	8-323
0x3424.0100	BPCL Port 1 (Best Packet Count Limit)	R/W	0x0000.03FF	8-324
0x3424.0100 + 4 * (Port Number – 1)	BPCL Port 2–26 (Best Packet Count Limit)	R/W	0x0000.03FF	8-325
0x3424.0168	GBTR (Global Broadcast Timer Register)	R/W	0x00xx.xxxx	8-326
0x3424.030C	OFCEN (Output Flow Control Enable Register)	R/W	0x02FF.FFFE	8-327
0x3424.0200	TDAD (Port 1 Transmit Descriptor Aging Discard Count Register)	R/W	0x0000.0000	8-328
0x3424.0200 + 4 * (Port Number – 1)	TDAD (Port 2–26 Transmit Descriptor Aging Discard Count Register)	R/W	0x0000.0000	8-329
0x3424.0400	Port 1 Low Priority Start and End Address)	R/W	0x007F.0000	8-330

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3424.0500–0x3424.1D00	Port 2–26 Low Priority Start and End Address	R/W	0x107F.1000, 0x017F.0100, 0x117F.1100, 027F.0200, 0x127F.1200, 0x037F.0300, 0x137F.1300, 0x047F.0400, 0x147F.1400, 0x057F.0500, 0x157F.0500, 0x157F.1500, 0x067F.0600, 0x167F.1600, 0x077F.0700, 0x177F.1700, 0x087F.0800, 0x187F.1800, 0x097F.0900, 0x0A7F.0A00, 0x1A7F.1A00, 0x0B7F.0B00, 0x1B7F.1B00, 0x0CFF.0C00, 0x1CFF.1C00	8-331
0x3424.040C	Port 1 High Priority Start and End Address	R/W	0x00FF.0080	8-332

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3424.050C– 0x3424.1D0C	Port 2–26 High Priority Start and End Addresses	R/W	0x10FF.1000, 0x01FF.0100, 0x11FF.1100, 0x02FF.0200, 0x12FF.1200, 0x03FF.0300, 0x13FF.1300, 0x04FF.0400, 0x14FF.1400, 0x05FF.0500, 0x15FF.1500, 0x06FF.0600, 0x16FF.1600, 0x07FF.0700, 0x17FF.1700, 0x08FF.0800, 0x18FF.1800, 0x09FF.0900, 0x19FF.1900, 0x0aFF.0a00, 0x1AFF.1A00, 0x0BFF.0B00, 0x1BFF.1B00, 0x0DFF.0D00, 0x1DFF.1D00	8-333
0x3424.0404	Port 1 Low Priority Read and Write Pointer)	R/W	0x4000.0000	8-334

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3424.0504–0x3424.1D04	Port 2–26 Low Priority Read and Write Pointers	R/W	0x5000.1000, 0x4100.0100, 0x5100.1100, 0x4200.0200, 0x5200.0200, 0x4300.0300, 0x5300.1300, 0x4400.0400, 0x5400.1400, 0x4500.0500, 0x5500.1500, 0x4600.0600, 0x5600.1600, 0x4700.0700, 0x5700.1700, 0x4800.0800, 0x5800.1800, 0x4900.0900, 0x5900.1900, 0x4A00.0A00, 0x5A00.1A00, 0x4B00.0B00, 0x5B00.1B00, 0x4C00.0C00, 0x5C00.1C00	8-336
0x3424.0410	Port 1 High Priority Read and Write Pointer	R/W	0x4080.0080	8-338

Table 8.1 Register Summary

Register Offset	Register Name	CPU R/W	Default	Page
0x3424.0510–0x3424.1D10	Port 2–26 High Priority Read and Write Pointers	R/W	0x5080.1080, 0x4180.0180, 0x5180.1180, 0x4280.0280, 0x5280.1280, 0x4380.0380, 0x5380.1380, 0x4480.0480, 0x5480.1480, 0x4580.0580, 0x5580.1580, 0x4680.0680, 0x5680.1680, 0x4780.0780, 0x5780.1780, 0x4880.0880, 0x5880.1880, 0x4980.0980, 0x5980.1980, 0x4A80.0A80, 0x5A80.1A80, 0x4B80.0B80, 0x5B80.1B80, 4C80.0C80, 0x5C80.1C80	8-340
0x3424.0408	Port 1 Low Priority Descriptor Peak and Ring Counter	R	0x0000.0000	8-342
0x3424.0508–0x3424.1D08	Port 2–26 Low Priority Descriptor Peak and Ring Counters	R	0x0000.0000	8-343
0x3424.0414	Port 1 High Priority Descriptor Peak and Ring Counter	R	0x0000.0000	8-344
0x3424.0514–0x3424.1D14	Ports 2–26 High Priority Descriptor Peak and Ring Counters	R	0x0000.0000	8-345
0x3424.0418	OFC1 (Port 1 Output Flow Control Register)	R/W	0x0000.0000	8-346
0x3424.0518–0x3424.1D18	OFC 2–26 (Port 2–26 Output Flow Control Registers)	R/W	0x0000.0000	8-348
0x3424.4000	Receive Descriptor Memory	R/W	0x0000.0000	8-350

Table 8.1 Register Summary

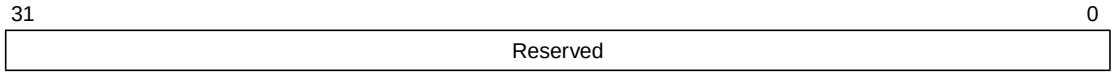
Register Offset	Register Name	CPU R/W	Default	Page
0x3424.8000–0x3424.B7FC	Descriptor Memory Group 1	R/W	0xxxxx.xxxx	8-353
0x3424.C000–0x3424.F7FC	Descriptor Memory Group 2	R/W	0xxxxx.xxxx	8-355
0x3421.0000	MIIM_CTL1	R/W	0x0000.0000	8-358

8.1 Global Registers: Base Address 0x3420.0000

The registers in this section are for general (global) ASIC control and do not specifically apply to a particular port.

Register: 0x3420.0000

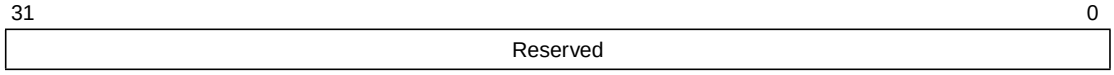
RESERVED
CPU:R/W
Default:0x0000.0000



Reserved	**Reserved** These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.	**[31:0]**

Register: 0x3420.0004

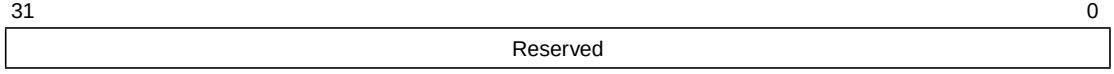
RESERVED
CPU:R/W
Default:0x0000.0000



Reserved	**Reserved** These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.	**[31:0]**

Register: 0x3420.0008

RESERVED
CPU:R/W
Default:0x0000.0000

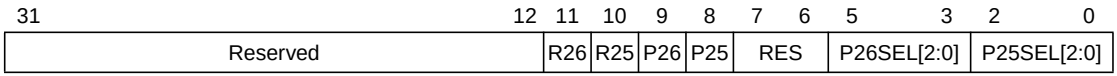


Reserved	**Reserved** These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.	**[31:0]**

Register: 0x3420.000C

MAC_SELECT
CPU:R/W
Default:0x0000.0300

Ports 25 and 26 (the uplink ports) can either use a 10/100 Mbits/s MAC or a 1000 Mbits/s MAC. Software must read the GPIO transceiver ID pins and, for 100BASE-T/1000BASE-T interfaces, read the AutoNegotiation result, then select the appropriate MAC using the MAC_SELECT register. When changing MAC selections, disable the port using the PORT_STP_LO and STP_PORT_HI registers of the switch engine (see the subsection entitled “PORT_STP_LO” on page 8-28 and the subsection entitled “PORT_STP_HI” on page 8-34).



Reserved **Reserved** **[31:12]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

R26 **RESET26** **11**
R26 is an individual RESET bit for the port 26 uplink 1000 Mbits/s module. At power-up and system reset, the default value of this bit is 0, which resets the associated module. It is the responsibility of software to set the bit to deactivate the reset.

R25 **RESET25** **10**
R25 is an individual RESET bit for the port 25 uplink 1000 Mbits/s module. At power-up and system reset, the default value of this bit is 0, which resets the associated module. It is the responsibility of software to set the bit to deactivate the reset.

P26 **PWR_DOWN/EWRAP IN TBI** **9**
In TBI mode, the P26 bit directly drives the SYNC/TXCKP pin for port 26 (pin V2). In the SMII, GMII, and MII modes, this bit is ignored. The default value of this bit is 1.

P25 **PWR_DOWN/EWRAP IN TBI** **8**

In TBI mode, the P25 bit directly drives the SYNC/TXCKP pin for port 25 (pin F2). In SMII, GMII and MII modes this bit is ignored. The default value of this bit is 1.

RES **Reserved** **[7:6]**

These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

P26SEL[2:0] **Port 26 Select** **[5:3]**

These bits configure uplink port 26. The configurations supported are shown in the table.

P26SEL[2:0]	Description
0b000	All uplink outputs are put in high impedance state (default value)
0b001	10/100 Mbits/s MAC, SMII
0b101	10/100 Mbits/s MAC, MII
0b110	1 Gbit/s MAC, GMII
0b111	1 Gbit/s MAC, TBI

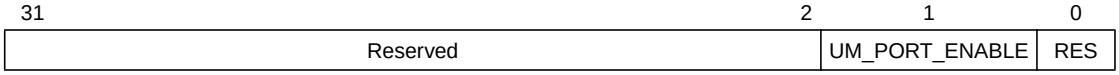
P25SEL[2:0] **Port 25 Select** **[2:0]**

These bits configure uplink port 25. The configurations supported are shown in the table.

P26SEL[2:0]	Description
0b000	All uplink outputs are put in high impedance state (default value)
0b001	10/100 Mbits/s MAC, SMII
0b101	10/100 Mbits/s MAC, MII
0b110	1 Gbit/s MAC, GMII
0b111	1 Gbit/s MAC, TBI

Register: 0x3420.0010

GENERAL_PURPOSE
CPU:R/W
Default:0x0000.0002



Reserved **Reserved** **[31:2]**

These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

UM_PORT_ENABLE

Unknown Multicast Port Enable **1**

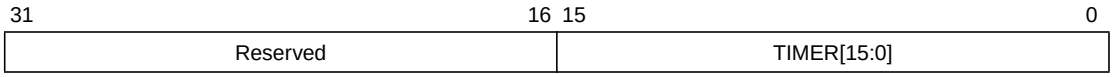
UM_PORT_ENABLE	Description
0	Normal (default)
1	Unknown packets are forwarded to the destination ports or to the CPU, as appropriate.

RES **Reserved** **0**

These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

Register: 0x3420.0014

GLOBAL_TIMER
CPU:R/W
Default:0x000.0271



Reserved **Reserved** **[31:16]**

These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

TIMER[15:0] **Prescaler Timer** **[15:0]**

The TIMER[15:0] output is used as the prescaler for the timer in the Switch Engine to remove an entry after some period of time and for the Transmit Descriptor to remove a packet if it is not sent out to the destination port within one second.

The TIMER[15:0] value is decremented once every 16 ns (one 62.5 MHz clock cycle). The default value of 0x0271 yields a value of 10 µs, as shown:

0x271 (625 decimal) * 16 ns = 10,000 ns = 10 µs.

The default value of TIMER[15:0] should be changed only when the clock frequency to the ARM CPU changes, such that the 10 µs timer period is maintained.

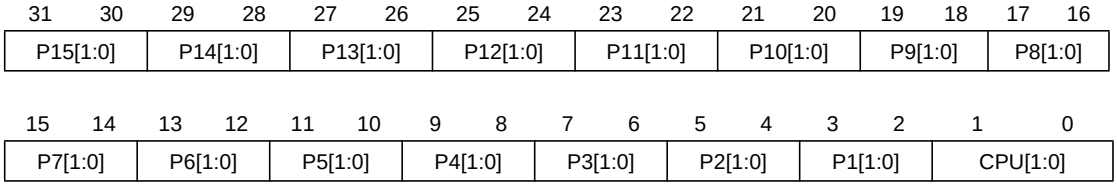
Register: 0x3420.0018

PORT_STP_LO
CPU:R/W
Default:0x0000.0000

To implement the spanning tree protocol, each port can be put into several modes that control which, if any, packets flow through the port, and whether learning takes place. The modes are:

- **Disabled**: No packets are received from or transmitted out of the port. No learning takes place for packets received from the port. No packets may be buffered to this port.
- **Blocking**: The switching engine filters Unicast packets. Multicast packets are also filtered, unless the “EMP” bit is set for the multicast address entry in the ARL. The CPU may transmit packets out of the port when it forces a port mask. No learning takes place for packets received from the port. Packets from other ports are not forwarded out of this port.
- **Listening:** An intermediate state used for timing purposes by the spanning tree software protocol. From the hardware perspective, this is the same as blocking.
- **Learnin**g: All packets are used for source address learning. The switching engine filters Unicast packets. Multicast packets are also filtered, unless the “EMP” bit is set for the multicast address entry in the ARL. The CPU may transmit packets out of the port when it forces a port mask. Packets from other ports are not forwarded out of this port.
- **Forwarding**: The port receives packets, and forwards them as appropriate. Learning may also take place.

The spanning tree state registers (PORT_STP_LO and PORT_STP_HI) control how the switch engine transmits and receives frames on the port. All ports of a trunked link should be programmed with the same spanning tree port state. If this is not done (for example, during the brief time between writing the two configuration words), packets may be filtered inconsistently, depending on which trunk link is chosen for that particular packet. Ports are enabled and disabled within the system using the PORT_STP_LO and PORT_STP_HI registers.



P15[1:0] **Port 15 Port Spanning Tree Protocol [31:30]**

The P15[1:0] bits program the Spanning Tree Protocol for port 15.

P15[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P14[1:0] **Port 14 Port Spanning Tree Protocol [29:28]**

The P14[1:0] bits program the Spanning Tree Protocol for port 14.

P14[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P13[1:0] **Port 13 Port Spanning Tree Protocol [27:26]**

The P13[1:0] bits program the Spanning Tree Protocol for port 13.

P13[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P12[1:0] **Port 12 Port Spanning Tree Protocol [25:24]**
The P12[1:0] bits program the Spanning Tree Protocol for port 12.

P12[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P11[1:0] **Port 11 Port Spanning Tree Protocol [23:22]**
The P11[1:0] bits program the Spanning Tree Protocol for port 11.

P11[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P10[1:0] **Port 10 Port Spanning Tree Protocol [21:20]**
The P10[1:0] bits program the Spanning Tree Protocol for port 10.

P10[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P9[1:0] **Port 9 Port Spanning Tree Protocol [19:18]**
The P9[1:0] bits program the Spanning Tree Protocol for port 9.

P9[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P8[1:0] **Port 8 Port Spanning Tree Protocol [17:16]**
The P8[1:0] bits program the Spanning Tree Protocol for port 8.

P8[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P7[1:0] **Port 7 Port Spanning Tree Protocol [15:14]**
The P7[1:0] bits program the Spanning Tree Protocol for port 7.

P7[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P6[1:0] **Port 6 Port Spanning Tree Protocol [13:12]**
The P6[1:0] bits program the Spanning Tree Protocol for port 6.

P6[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P5[1:0] **Port 5 Port Spanning Tree Protocol [11:10]**
The P5[1:0] bits program the Spanning Tree Protocol for port 5.

P5[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P4[1:0] **Port 4 Port Spanning Tree Protocol [9:8]**

The P4[1:0] bits program the Spanning Tree Protocol for port 4.

P4[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P3[1:0] **Port 3 Port Spanning Tree Protocol [7:6]**

The P3[1:0] bits program the Spanning Tree Protocol for port 3.

P3[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P2[1:0] **Port 2 Port Spanning Tree Protocol [5:4]**

The P2[1:0] bits program the Spanning Tree Protocol for port 2.

P2[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P1[1:0] **Port 1 Port Spanning Tree Protocol [3:2]**

The P1[1:0] bits program the Spanning Tree Protocol for port 1.

P1[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

CPU[1:0] **CPU Port Spanning Tree Protocol [1:0]**

The CPU[1:0] bits program the Spanning Tree Protocol. The only valid values for the CPU port are Disabled (0b00) and Forward (0b11).

P1[1:0]	Description
0b00	Disable CPU Port (default)
0b01	Reserved
0b10	Reserved
0b11	Forward

Note: Packets *can* be mirrored from a port in the blocking or learning state.

Register: 0x3420.001C

PORT_STP_HI
CPU:R/W
Default:0x0000.0000

31 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
Reserved	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16

Reserved **Reserved** **[31:22]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

P26[1:0] **Port 26 Port Spanning Tree Protocol** **[21:20]**
The P26[1:0] bits program the Spanning Tree Protocol for port 26.

P26[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P25[1:0] **Port 25 Port Spanning Tree Protocol** **[19:18]**
The P25[1:0] bits program the Spanning Tree Protocol for port 25.

P25[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P24[1:0] **Port 24 Port Spanning Tree Protocol** **[17:16]**
The P24[1:0] bits program the Spanning Tree Protocol for port 24.

P24[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P23[1:0] **Port 23 Port Spanning Tree Protocol [15:14]**
The P23[1:0] bits program the Spanning Tree Protocol for port 23.

P23[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P22[1:0] **Port 22 Port Spanning Tree Protocol [13:12]**
The P22[1:0] bits program the Spanning Tree Protocol for port 22.

P22[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P21[1:0] **Port 21 Port Spanning Tree Protocol [11:10]**
The P21[1:0] bits program the Spanning Tree Protocol for port 21.

P21[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P20[1:0] **Port 20 Port Spanning Tree Protocol [9:8]**
The P20[1:0] bits program the Spanning Tree Protocol for port 20.

P20[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P19[1:0] **Port 19 Port Spanning Tree Protocol [7:6]**
The P19[1:0] bits program the Spanning Tree Protocol for port 19.

P19[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P18[1:0] **Port 18 Port Spanning Tree Protocol [5:4]**
The P18[1:0] bits program the Spanning Tree Protocol for port 18.

P18[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

P17[1:0] **Port 17 Port Spanning Tree Protocol [3:2]**
The P17[1:0] bits program the Spanning Tree Protocol for port 17.

P17[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

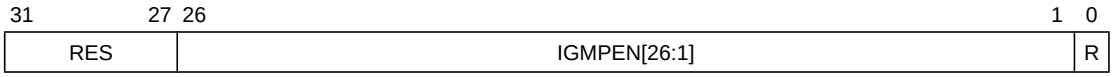
P16[1:0] **Port 16 Port Spanning Tree Protocol [1:0]**
The P16[1:0] bits program the Spanning Tree Protocol for port 16.

P16[1:0]	Description
0b00	Disable Port (default)
0b01	Blocking
0b10	Learning
0b11	Forward

Note: Packets can be mirrored from a port in the blocking or learning state.

Register: 0x3420.0020

IGMP_ENABLE
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:27]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

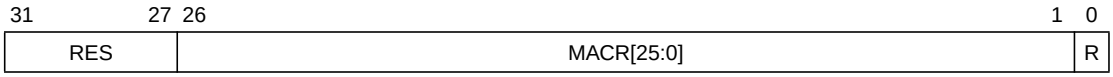
IGMPEN[25:0] **IGMP Capture Enable** **[26:1]**
The IGMPEN[26:1] bits enable IGMP (Internet Group Management Protocol) on ports 26 to 1. When the bit for a port is set, the corresponding port snoops on receive traffic and flags IGMP/OSPF/PIM message packets to be forwarded to the CPU.

The two-bit IGMP field in the data from an input packet sent to the switch engine for processing indicates whether the received packet is an IGMP, PIM, or OSPF protocol packet. The port logic creates this field and the the switch engine uses it to create the forwarding mask.

R **Reserved** **0**
This bit is reserved. Software should write 0 to this bit and assume it is undefined for reads.

Register: 0x3420.0024

MAC_RESET
CPU:R/W
Default:0x0000.0000



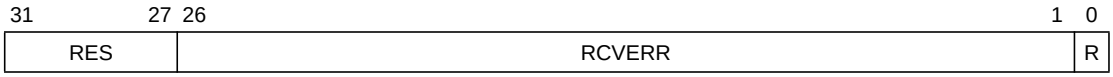
RES **Reserved** **[31:27]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

MACR[25:0] **MAC Reset** **[26:1]**
When a bit is set, a software reset is performed on the corresponding MAC. This bit should be used only when software wishes to do an individual port reset. The default value for each bit is 0.

R **Reserved** **0**
This bits is reserved. Software should write 0 to this bit and assume it is undefined for reads.

Register: 0x3420.0028

RCV_ERR
CPU:R/W
Default:0x000.0000



RES **Reserved** **[31:27]**
These bits are reserved. Software should write 0 to these bits and assume it is undefined for reads.

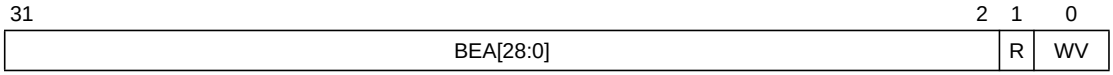
RCVERR[25:0]
Receive Error Packets **[26:1]**
When a bit is set, the corresponding port is enabled to receive error packets equal to or greater than 64 bytes (bits 26 to 1 correspond to ports 26 to 1).

R **Reserved** **0**
This bit is reserved. Software should write 0 to this bit and assume it is undefined for reads.

Register: 0x3420.002C

BUS_ERR
CPU:R
Default:0x0000.0000

To assist system debugging, the CPU must have access to registers that provide access to key signals or important state machines for each module within the ASIC. The Bus Error detector monitors each cycle and terminates any cycle that requires more than 512 clocks or that attempts to access any protected or nonexistent memory regions. Before it terminates the cycle, it supplies a Bus Error Interrupt to the processor and latches the address that caused the error. The BUS_ERR register holds the address and status of the latest bus error. The word address is saved as well as the read/write status.



BEA[29:0] **Bus Error Address** **[31:2]**
The BEA[29:0] bits contain the address of the last cycle to cause a bus error on the Advanced System Bus (ASB).

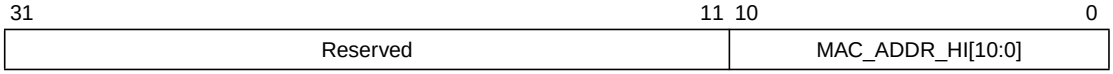
R **Reserved** **1**
This bit is reserved. Software should write 0 to this bit and assume it is undefined for reads.

WV **Write Value** **0**
The WV bit shows the read/write status value at the time of a bus error.

WV	Description
0	Read
1	Write

Register: 0x3420.0030

MAC_ADDR1
CPU:R/W
Default:0x0000.0xxx



Reserved **Reserved** **[31:11]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

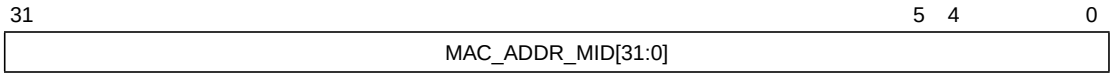
MAC_ADDR_HI[10:0]
Upper MAC Address Bits for All Ports[10:0]
These bits contain the upper 11 bits [47:37] of the 48-bit MAC address for all ports. The next 32 bits of the 48 bits are in the MAC_ADDR_MID field (see the subsection entitled “MAC_ADDR2” on page 8-42).

The lower 5 bits of the 48-bit MAC address are hardwired in the port logic to the port number for each individual port.

The upper 43 bits of the 48-bit address of incoming packets are compared against the 43 bits in the MAC_ADDR1 and MAC_ADDR2 registers. The lower 5 bits of the incoming address are also compared against the hardwired address of each port. If all 48 bits match, the packet is destined for that port.

Register: 0x3420.0034

MAC_ADDR2
CPU:R/W
Default:0xxxxx.xxxx



MAC_ADDR_MID[31:0]

Middle MAC Address Bits for All Ports [31:0]

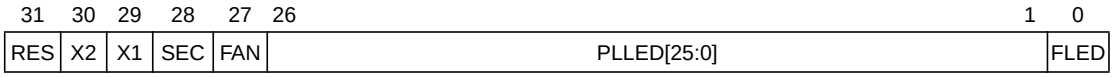
These bits contain the middle 32 bits [36:5] of the 48-bit MAC address for all ports. The upper 11 bits of the 48 bits are in the MAC_ADDR_HI field (see the subsection entitled “MAC_ADDR1” on page 8-41).

The lower 5 bits of the 48-bit MAC address are hardwired in the port logic to the port number for each individual port.

The upper 43 bits of the 48-bit address of incoming packets is compared against the 43 bits in the MAC_ADDR1 and MAC_ADDR2 registers. The lower 5 bits of the incoming address is also compared against the hardwired address of each port. If all 48 bits match, the packet is destined for that port.

Register: 0x3420.0038

LED_DATA1
CPU:R/W
Default:0xFFFF.FFFF



RES **Reserved** **31**

This bit is reserved. Software should write 0 to this bit and assume it is undefined for reads.

X2 **Transceiver 2 Insertion** **30**

When this bit is 1, it drives the uplink port 26. Transceiver 2 LED if bits [29:28] of the LED_CNTRL_1B register are 0b00.

The Transceiver 2 LED lights only if the inserted PHY transceiver module for uplink port 26 is valid for the product, as indicated by the transceiver ID, and is completely seated. The CPU is responsible for reading the transceiver ID bits and then appropriately programming the LED_DATA1 and LED_CTRL_1B registers.

X1 **Transceiver 1 Insertion** **29**

When this bit is 1, it drives the uplink port 25. Transceiver 1 LED if bits [27:26] of the LED_CNTRL_1B register are 0b00.

The Transceiver 1 LED lights only if the inserted PHY transceiver module for uplink port 25 is valid for the product, as indicated by the transceiver ID, and is completely seated. The CPU is responsible for reading the transceiver ID bits and then appropriately programming the LED_DATA1 and LED_CTRL_1B registers.

SEC **Security** **28**

When this bit is 1, it drives the Security LED if bits [25:24] of the LED_CNTRL_1B register are 0b00.

Software controls the Security LED; it is not controlled directly from any hardware source. When the CPU detects a security violation, the CPU may turn this LED

on by means of the LED_DATA_1 register, provided bits SEC[1:0] in the LED_CTRL_1B register are 0b00. Or, the Security LED can be controlled from Timer 0 (bits SEC[1:0] in the LED_CTRL_1B register are 0b01) or Timer 1 (bits SEC[1:0] in the LED_CTRL_1B register are 0b10).

FAN **FAN** **27**

When this bit is 1, it drives the Fan LED if bits [23:22] of the LED_CNTRL_1B register are 0b00.

There may be one or more fans present in the system. All have open drain outputs that go LOW when a locked fan rotor condition occurs. The outputs may be wire OR'd together and connected to a dedicated input on the ASIC. This input can also be driven from a temperature sensor for a more accurate indication of trouble.

The Fan LED and the System Fault LED flash in unison at the rate programmed into Timer 0 or Timer 1, depending on the programming of the Fan control multiplexer, which is programmed through the LED_CTRL_1B register. Software can also read the Fan status from the GPIO2 Status register.

PLLED[25:0] **Port Link Status LED** **[26:1]**

These bits, when 1, drive the corresponding Link Status LEDs if the appropriate control bits in the LED_CNTRL_1A and LED_CNTRL_1B registers are 0b00.

For example, bit 26 drives the Port 26 Link LED when bits [21:20] of the LED_CNTRL_1B register are 0b00.

FLED **Fault LED** **0**

This bit directly controls the Fault LED, provided the FLT[1:0] bits in the LED_CTRL_1A register are 0b00.

Register: 0x3420.003C

LED_DATA2
CPU:R/W
Default:0xFFFF.FFFF

31	30	29	28	27	26 ... 1	0
MDIX	ER	FDX	SPD	ACT	PTMODE[25:0]	ST

MDIX **MDIX Port Mode LED** **31**

If this bit is set it illuminates the MDIX LED if the hardware multiplexer is configured to select the LED_DATA2 register as the source that drives the LEDs. The per port Mode LEDs then reflect the MDIX mode of the port.

If a port has been configured or has autoconfigured to MDI, the LED is illuminated for that port. MDIX corresponds to the default or standard connection of the transmit and receive signal configuration of a switch port to its PHY. MDI corresponds to the standard configuration of the transmit and receive signals of an end node to its PHY.

In this mode of operation, the hardware control multiplexer is not important, because the CPU controls this mode. The CPU continually polls the PHYs to determine the MDI interface state. If the link is found to be in the MDIX state, the corresponding PTMODE[25:0] bit in this register is set. It is not likely nor is it the intent of this mode to reflect exactly the dynamic state of the MDIX mode of each interface.

This bit drives the MDIX LED only if bits [31:30] of the LED_CNTRL_2B register are 0b00.

ER **Error Mode LED** **30**

If the Error display mode has been selected (using the Mode Select switch), this bit is 1. It illuminates the Error mode LED if the hardware multiplexer is configured to select the LED_DATA2 register as the source that drives the LEDs. The per port Mode LEDs then reflect the error mode of the port.

An error is defined as any event that results in a CRC error or a jabber packet. When an error occurs, the

corresponding PTMODE[25:0] bit in this register is set, and the port Mode LED illuminates for 5 ms. This 5 ms pulse stretch function is not retriggerable during the 5 ms stretch interval. Immediately after the 5 ms duration, the LED is extinguished. The LED may then illuminate again for any error event on the given port.

This bit drives the Error Mode LED only if bits [29:28] of the LED_CNTRL_2B register are 0b00.

FDX **FDX Mode LED** **29**

If the FDX display mode has been selected (using the Mode Select switch), this bit is 1. It illuminates the FDX mode LED if the hardware multiplexer is configured to select the LED_DATA2 register as the source that drives the LEDs. The per port Mode LEDs then reflect the full-duplex mode of the port.

When this bit is 1, it drives the FDX Mode LED and a per-port Mode LED is illuminated if the port is in the Full-Duplex mode. If the port is found to be in the full-duplex state, the corresponding PTMODE[25:0] bit in this register is set.

This bit drives the FDX Mode LED only if bits [27:26] of the LED_CNTRL_2B register are 0b00.

SPD **SPD Mode LED** **28**

If the SPD display mode has been selected (using the Mode Select switch), this bit is 1. It illuminates the SPD mode LED if the hardware multiplexer is configured to select the LED_DATA2 register as the source that drives the LEDs. The per port Mode LEDs then reflect the speed mode of the port.

When this bit is 1, it drives the SPD mode LED, and the per-port PTMODE[25:0] bit in this register is set to illuminate the corresponding Mode LED if the given port is operating at its fastest speed.

10/100 Mbits/s port Mode LEDs illuminate only when the port has been configured to or has AutoNegotiated to 100 Mbits/s operation. 100/1000 Mbits/s port Mode LEDs illuminate only when the port has been configured to or has AutoNegotiated to 1000 Mbits/s operation.

This bit drives the SPD Mode LED only if bits [25:24] of the LED_CNTRL_2B register are 0b00.

ACT **ACT Mode LED** **27**

If the ACT display mode has been selected (using the Mode Select switch), this bit is 1. It illuminates the ACT mode LED if the hardware multiplexer is configured to select the LED_DATA2 register as the source that drives the LEDs. The per port Mode LEDs then reflect the activity mode of the port.

When this bit is 1, it drives the ACT Mode LED and the corresponding per-port PTMODE[25:0] bit in this register is set to illuminate the corresponding Mode LED if the port is transmitting or receiving packets.

Activity is defined as sending or receiving any good or bad packet. When activity is detected, the Mode LED, while configured to indicate activity, illuminates for a duration of 5 ms. This 5 ms pulse stretch function is not retriggerable during the 5 ms stretch interval. Immediately after the 5 ms duration, the LED is extinguished. The LED may then illuminate again for any activity event on the given port.

There is no specified off time. Pause frame transmission and reception is also considered to be an activity event. A disabled port should not reflect receive activity.

This bit drives the ACT mode LED only if bits [23:22] of the LED_CNTRL_2B register are 0b00.

PTMODE[25:0] **Port Mode LED** **[26:1]**

These bits drive the corresponding Port Mode LEDs if the appropriate control bits in the LED_CNTRL_2A and LED_CNTRL_2B registers are 0b00.

For example, bit 26 drives the Port 26 Mode LED when bits [21:20] of the LED_CNTRL_2B register are 0b00.

The LED meanings for each mode are shown in the table.

Mode	Bit State	Description
ER	0	No errors
	1	For each CRC error or jabber packet detected, the LED illuminates for 5 ms.
FDX	0	Port is operating at half duplex
	1	Port is operating at full duplex
SPD	0	Port is operating at lower speed
	1	Port is operating at higher speed
ACT	0	No network traffic exists on the given port or the port is disabled
	1	Network traffic exists on the given Port

ST **Self Test LED** **0**

At power-up and reset, this LED is illuminated and remains so until the software and hardware self-test sequences complete successfully. This LED has no hardware function source.

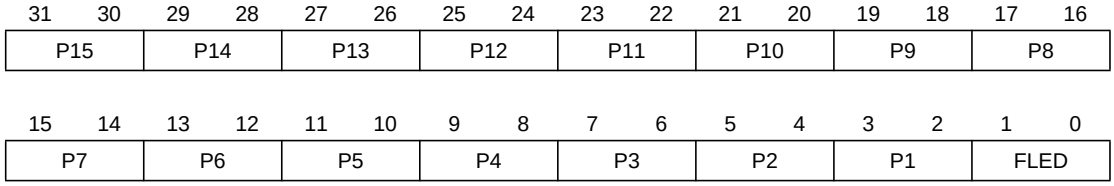
The ST bit drives the Self-test LED directly provided the ST[1:0} bits in the LED_CTRL_2A register are 0b00.

Register: 0x3420.0040

LED_CNTRL_1A
CPU:R/W
Default:0x0000.0000

The LED_CNTRL_1A LED control register allows software to select the source data that drives certain LEDs. There are four possible sources:

- The LED_DATA1 Register
- Timer 0
- Timer 1
- Hardware Control



P15[1:0] **Port 15 Link LED Control [31:30]**
This two-bit field controls the port 15 multiplexer that selects the source that drives the port 15 Link LED.

P15[1:0]	LED Source
0b00	LED_DATA1 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

P14[1:0] to P1[1:0]
Port 14 to Port 1 Link LEDs Control [29:28] to [3:2]
These two-bit fields operate in a manner similar to that of P15[1:0] for the respective ports.

FLED[1:0] **Fault LED Control [1:0]**
This two-bit field controls the Fault LED multiplexer that selects the source that drives the Fault LED.

FLED[1:0]	LED Source
0b00	LED_DATA1 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Not Used

Register: 0x3420.0044

LED_CNTRL_1B
CPU:R/W
Default:0x0000.0000

The LED_CNTRL_1B LED control register allows software to select the source data that drives certain LEDs. There are four possible sources:

- The LED_DATA1 Register
- Timer 0
- Timer 1
- Hardware Control

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
RES	X2	X1	SEC	FAN	P26	P25	P24

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
P23	P22	P21	P20	P19	P18	P17	P16

RES **Reserved** **[31:30]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

X2[1:0] **Transceiver 2 Insertion Control** **[29:28]**
This two-bit field controls the Transceiver 2 multiplexer that selects the source that drives the Transceiver 2 LED.

X2[1:0]	LED Source
0b00	LED_DATA1 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	TIMER 1

X1[1:0] **Transceiver 1 Insertion Control** **[27:26]**

This two-bit field controls the Transceiver 1 multiplexer that selects the source that drives the Transceiver 1 LED.

X1[1:0]	LED Source
0b00	LED_DATA1 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	TIMER 1

SEC[1:0] **Security Control** **[25:24]**

This two-bit field controls the Security multiplexer, which determines whether the software bits (LED_DATA1 register), LED_TIMER0, or LED_TIMER1 drives the Security LED.

SEC[1:0]	LED Source
0b00	LED_DATA1 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	TIMER 1

FAN[1:0] **Fan Control** **[23:22]**

This two-bit field controls the Fan multiplexer, which determines whether the software bits (LED_DATA1 register), LED_TIMER0, LED_TIMER1, or the hardware drives the Fan LED.

X1[1:0]	LED Source
0b00	LED_DATA1 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

P26[1:0] **Port 26 Link LED Control** **[21:20]**

This two-bit field controls the port 26 Link multiplexer, which determines whether the software bits (LED_DATA1 register), LED_TIMER0, LED_TIMER1, or the hardware drives the port 26 Link LED.

P26[1:0]	LED Source
0b00	LED_DATA1 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

P25[1:0] to P16[1:0]

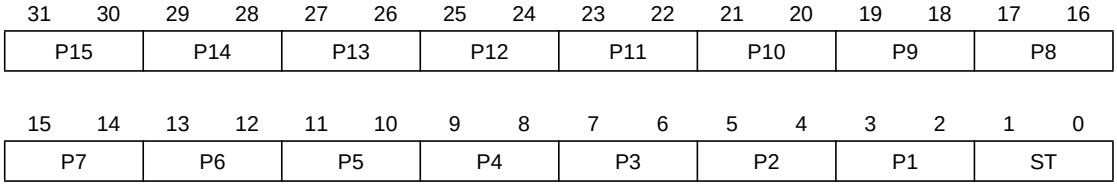
Port 25 to Port 16 Link LEDs Control [19:18] to [1:0]
These two-bit fields operate in a manner similar to that of P26[1:0] for the respective ports.

Register: 0x3420.0048

LED_CNTRL_2A
CPU:R/W
Default:0x0000.0000

The LED_CNTRL_2A LED control register allows software to select the source data that drives certain LEDs. There are four possible sources:

- The LED_DATA2 Register
- Timer 0
- Timer 1
- Hardware Control



P15[1:0] **Port 15 Mode LED Control [31:30]**
This two-bit field controls the port 15 multiplexer that selects the source that drives the port 15 Mode LED.

P15[1:0]	LED Source
0b00	LED_DATA2 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

P14[1:0] to P1[1:0]
Port 14 to Port 1 Mode LEDs Control [29:28] to [3:2]
These two-bit fields operate in a manner similar to that of P15[1:0] for the respective ports.

ST[1:0] **Self Test Control [1:0]**
This two-bit field controls the Self-Test multiplexer that selects the source that drives the Self-Test LED.

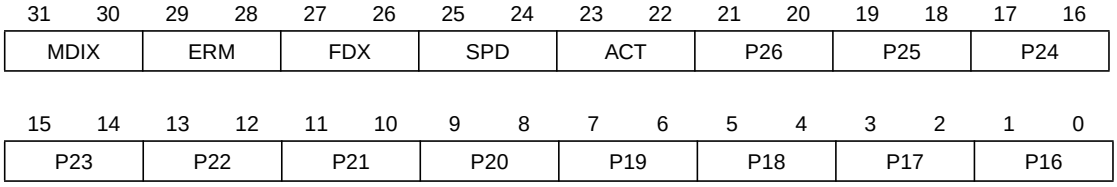
ST[1:0]	LED Source
0b00	LED_DATA2 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

Register: 0x3420.004C

LED_CNTRL_2B
CPU:R/W
Default:0x0000.0000

The LED_CNTRL_2B LED control register allows software to select the source data that drives certain LEDs. There are four possible sources:

- The LED_DATA2 Register
- Timer 0
- Timer 1
- Hardware Control



MDIX[1:0] **MDIX LED Control** **[31:30]**
This two-bit field controls the MDIX multiplexer that selects the source that drives the MDIX LED.

MDIX[1:0]	LED Source
0b00	LED_DATA2 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

ERM[1:0] **Error Mode LED Control** **[29:28]**
This two-bit field controls the Error multiplexer that selects the source that drives the Error Mode LED.

ERM[1:0]	LED Source
0b00	LED_DATA2 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

FDX[1:0] **FDX Mode LED Control** **[27:26]**

This two-bit field controls the FDX multiplexer that selects the source that drives the FDX Mode LED.

FDX[1:0]	LED Source
0b00	LED_DATA2 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

SPD[1:0] **SPD Mode LED Control** **[25:24]**

This two-bit field controls the SPD multiplexer that selects the source that drives the SPD Mode LED.

SPD[1:0]	LED Source
0b00	LED_DATA2 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

ACT[1:0] **ACT Mode LED Control** **[23:22]**

This two-bit field controls the ACT multiplexer that selects the source that drives the ACT Mode LED.

ACT[1:0]	LED Source
0b00	LED_DATA2 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

P26[1:0] **Port 26 Mode LED Control** **[21:20]**

This two-bit field controls the port 26 multiplexer that selects the source that drives the port 26 Mode LED.

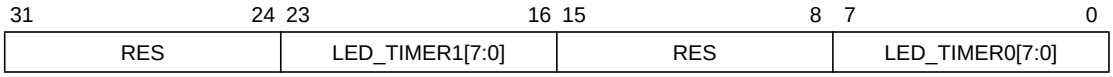
P26[1:0]	LED Source
0b00	LED_DATA2 Register
0b01	LED_TIMER0
0b10	LED_TIMER1
0b11	Hardware Control

P25[1:0] to P16[1:0]

Port 25 to Port 16 Mode LEDs Control [19:18] to [1:0]
These two-bit fields operate in a manner similar to that of P26[1:0] for the respective ports.

Register: 0x3420.0050

LED_TIMERS
CPU:R/W
Default:0x0026.0013



RES **Reserved** **[31:24]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

LED_TIMER1[7:0]
LED Timer 1 **[23:16]**
These bits provide an LED timer whose range is 16.7 ms to 4.29 seconds, according to the following formula:

Timer period = LED_TIMER1[7:0] * 2 * 8.39 ms.

A value of zero stops the timer. The LED_TIMER1[7:0] default value is 0x26, such that the default timer period is:

38 * 2 * 8.39 ms = 637.64 ms.

RES **Reserved** **[15:8]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

LED_TIMER0[7:0]
LED Timer 0 **[7:0]**
These bits provide an LED timer whose range is 16.7 ms to 4.29 seconds, according to the following formula:

Timer period = LED_TIMER0[7:0] * 2 * 8.39 ms.

A value of zero stops the timer. The LED_TIMER0[7:0] default value is 0x13, such that the default timer period is:

19 * 2 * 8.39 ms = 318.82 ms

Register: 0x3420.0054

LED_MODE
CPU:W/O
Default:0x0000.11FB

31 18	17 16	15 13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	GF	RES	JAB	OVR	UDR	OVS	NB	SE	AE	CRR	CRT	OVST	UDRT	EC	LC

RES **Reserved** **[31:18]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

GF[1:0] **Global Field** **[17:16]**
The Mode Switch input selects one of five display modes: ACT, FDX, SPD, ER, or MDIX. When the Mode Select switch is depressed, the currently illuminated Mode Select LED is turned off, and the next Mode Select LED in the sequence is illuminated.

When a given mode is selected, the individual Port Mode LEDs display the function selected from the depression of the Mode Select switch and the corresponding GF[1:0] bits as shown in the table are set to the mode selected.

The GF[1:0] bits select which of the four hardware sources is selected as the Mode LED input to the LED control multiplexer.

GF[1:0]	Hardware Source
0b00	Activity
0b01	Full Duplex
0b10	Speed
0b11	Error

The default state of GF[1:0] is 0b00.

RES **Reserved** **[15:13]**
These bits are reserved. Software should write 0 to these bits and assume they are undefined for reads.

JAB **Jabber RX_EN LED** **12**
The JAB bit (default = 1), in conjunction with bits [11:0] of this register control the global behavior of the Error Mode LED. This bit and the 12 bits that follow drive a

corresponding signal to the port logic. The port logic feeds the LED logic with the 13 signals, and the LED logic uses these signals to enable (HIGH) or mask (LOW) the corresponding error type. The LED logic OR's together the unmasked signals and uses the output to trigger the Error Mode LED stretch circuit. Any rising edge of the OR'ed output causes the event to generate a stretched 5 ms LED drive signal for the corresponding port. The 5 ms stretch operation provides a visible event to the user.

During the 5 ms pulse, additional assertions of the OR'ed error signal are ignored. Any one, or any combination, or all 13 error enable bits may be set. The bits control the Error Mode for all ports.

OVR	**Overrun RX_EN LED** Default = 0	**11**
UDR	**Undersize RX_EN LED** Default = 0	**10**
OVS	**Oversize RX_EN LED_** Default = 0	**9**
NB	**No Buffer RX_EN LED_** Default = 1	**8**
SE	**Symbol Error RX_EN LED_** Default = 1	**7**
AE	**LED Align Error RX_EN LED_** Default = 1	**6**
CRR	**LED CRC Error RX_EN LED_** Default = 1	**5**
CRT	**CRC Error TX_EN LED_** Default = 1	**4**
OVST	**Oversize TX_EN LED_** Default = 1	**3**
UDRT	**Underrun TX_EN LED** Default = 0	**2**

EC	**Excessive Collisions TX_EN LED** Default = 1	**1**
LC	**Late Collision TX_EN LED** Default = 1	**0**

This register is a write only register. Data written cannot be read back.

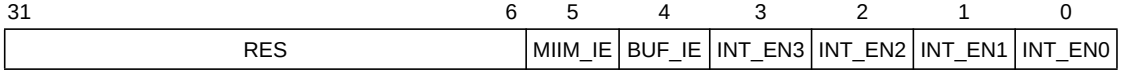
Register: 0x3420.0100

LAN_INT_EN
CPU:R/W
Default:0x0000.0000

The LAN_INT_EN register allows the LAN interrupts to be independently enabled for each receive queue, MIIM command completions, and the buffer available notification. The ARM interrupt controller must also be configured to enable interrupts. The four receive interrupts share a single receive interrupt signal (Network Rx) to the interrupt controller. The buffer available interrupt and MIIM interrupts are tied to the SWITCH interrupt signal. The raw SWITCH interrupt bit is active HIGH, and is determined as follows:

SWITCH = (MIIM_INT_EN and MIIM_INT_STAT) or (BUF_INT_EN and BUF_INT_STAT)

The transmit interrupt is controlled and monitored solely through the ARM interrupt controller. LAN_INT_EN controls whether or not the given event will cause an interrupt to occur. LAN_INT_STAT is set by the ASIC whenever the specified event occurs, regardless of LAN_INT_EN. LAN_INT_STAT is latched by the ASIC when the events occur, and it is cleared when the CPU writes a ‘1’ to the corresponding bit of LAN_INT_ACK. LAN_INT_ACK is used by the CPU to acknowledge the interrupt and clear the corresponding bit of LAN_INT_STAT.



RES **Reserved** **[31:6]**
These bits are reserved. Software should write a 0 to these bits. The bits return 0 when read.

MIIM_IE **MIIM Interrupt Enable** **5**

MIIE_IE	Description
0	Do not enable interrupt (default)
1	Enable an interrupt when the CPU-initiated MDIO/MDC commands complete (for example, when the CPU_MIIM_BUSY bit changes from 1 to 0)

BUF_IE **Buffer Interrupt Enable** **4**

BUF_IE	Description
0	Do not enable interrupt (default)
1	Enable an interrupt when a requested buffer becomes available (for example, when the VALID bit of the CPU_BUF_REQ register changes from 0 to 1).

INT_EN3 **CPU RX Priority 3 Queue Interrupt Enable** **3**

INT_EN3	Description
0	Do not enable interrupt (default)
1	Enable an interrupt for packets received on CPU priority 3 queue.

INT_EN2 **CPU RX Priority 2 Queue Interrupt Enable** **2**

INT_EN2	Description
0	Do not enable interrupt (default)
1	Enable an interrupt for packets received on CPU priority 2 queue.

INT_EN1 **CPU RX Priority 1 Queue Interrupt Enable** **1**

INT_EN1	Description
0	Do not enable interrupt (default)
1	Enable interrupt for packets received on CPU priority 1 queue.

INT_EN0 **CPU RX Priority 0 Queue Interrupt Enable** **0**

INT_EN0	Description
0	Do not enable interrupt (default)
1	Enable interrupt for packets received on CPU priority 0 queue.

Register: 0x3420.0104

LAN_INT_ACK
CPU:W
Default:0x0000.0000

31 … 6	5	4	3	2	1	0
RES	MIA	BIA	INT_ACK3	INT_ACK2	INT_ACK1	INT_ACK0

RES **Reserved** **[31:6]**

These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

MIA **MIIM Interrupt Acknowledge** **5**

MIIM_ACK	Description
0	No action (default)
1	Resets the MDIO/MDC management command complete interrupt and clears the MIIM_INT_STAT bit

BIA **Buffer Interrupt Acknowledge** **4**

BUF_ACK	Description
0	No action (default)
1	Resets the buffer available interrupt and clears the BUF_INT_STAT bit

INT_ACK3 **CPU RX Priority 3 Queue Interrupt Acknowledge** **3**

INT_ACK3	Description
0	No action (default)
1[1]	Resets the priority 3 receive queue interrupt and clears the RX_INT_STAT3 bit

1. To clear the interrupt, software must set the bit, then clear it.

INT_ACK2 **CPU RX Priority 2 Queue Interrupt Acknowledge** **2**

INT_ACK2	Description
0	No action (default)
1[1]	Resets the priority 2 receive queue interrupt and clears the RX_INT_STAT2 bit

1. To clear the interrupt, software must set the bit, then clear it.

INT_ACK1 CPU RX Priority 1 Queue Interrupt Acknowledge 1

INT_ACK1	Description
0	No action (default)
1[1]	Resets the priority 1 receive queue interrupt and clears the RX_INT_STAT1 bit

1. To clear the interrupt, software must set the bit, then clear it.

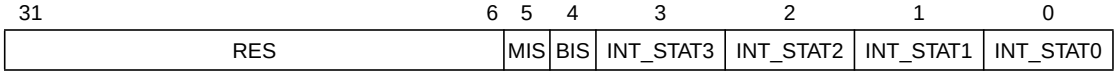
INT_ACK0 CPU RX Priority 0 Queue Interrupt Acknowledge 0

INT_ACK0	Description
0	No action (default)
1[1]	Resets the priority 0 receive queue interrupt and clears the RX_INT_STAT0 bit

1. To clear the interrupt, software must set the bit, then clear it.

Register: 0x3420.0108

LAN_INT_STAT
CPU:R
Default:0x0000.0000



RES **Reserved** **[31:6]**

These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

MIS **MIIM Interrupt Status** **5**

MIIM_STAT	Description
0	Command has not completed (default)
1	A CPU-initiated MDIO/MDC command has completed.

BIS **Buffer Interrupt Status** **4**

BUF_STAT	Description
0	Command has not completed (default)
1	A requested buffer has become available. Check the CPU_BUF_REQ register.

INT_STAT3 **CPU RX Priority 3 Queue Interrupt Status** **3**

INT_STAT3	Description
0	No packets received (default)
1	Packets have been received on the CPU priority 3 receive queue.

INT_STAT2 **CPU RX Priority 2 Queue Interrupt Status** **2**

INT_STAT2	Description
0	No packets received (default)
1	Packets have been received on the CPU priority 2 receive queue.

INT_STAT1 CPU RX Priority 1 Queue Interrupt Status 1

INT_STAT2	Description
0	No packets received (default)
1	Packets have been received on the CPU priority 1 receive queue.

INT_STAT0 CPU RX Priority 0 Queue Interrupt Status 0

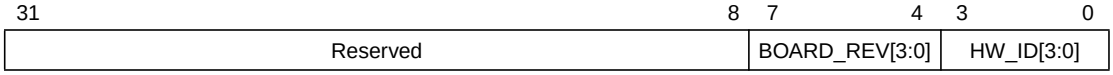
INT_STAT0	Description
0	No packets received (default)
1	Packets have been received on the CPU priority 0 receive queue.

8.2 GPIO Registers: Base Address 0x9000.0000

This section describes the GPIO registers. The GPIO registers can be used to read the state of input pins or write to output pins. The registers are flexibly assigned various functions.

Register: 0x9000.0000

GPIO1_STAT
CPU:R
Default:0x0000.0000



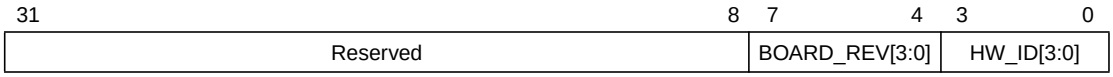
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

BOARD_REV[3:0]
Board Revision **[7:4]**
These bits are latched in during reset according to the values set with pull-up or pull-down resistors on the board revision pins. The BOARD_REV[3:0] input pins are multiplexed with data lines DATA[83:80], respectively. Software should never set these bits.

HW_ID[3:0] **Hardware ID** **[3:0]**
These bits are latched in during reset according to the values set with pull-up or pull-down resistors on the hardware ID pins. The HW_ID[3:0] input pins are multiplexed with data lines DATA[85, 84, 79, 78], respectively. Software should never set these bits.

Register: 0x9000.0004

GPIO1_RAW_STAT
CPU:R
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

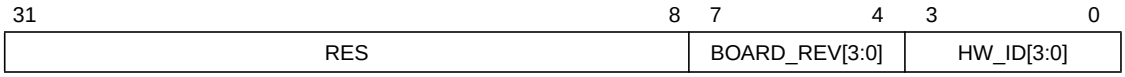
BOARD_REV[3:0]
Board Revision **[7:4]**
These bits are latched in during reset according to the values set with pull-up or pull-down resistors on the board revision pins. The BOARD_REV[3:0] input pins are multiplexed with data lines DATA[83:80], respectively. Software should never set these bits.

HW_ID[3:0] **Hardware ID** **[3:0]**
These bits are latched in during reset according to the values set with pull-up or pull-down resistors on the hardware ID pins. The HW_ID[3:0] input pins are multiplexed with data lines DATA[85, 84, 79, 78], respectively. Software should never set these bits.

HW_ID[3:0]	Description
0b0000	24 10/100 Mbits/s ports plus 2 1000 Mbits/s ports
0b0001	12 10/100 Mbits/s ports plus 2 1000 Mbits/s ports

Register: 0x9000.0008

GPIO1_EN_SET
CPU:R/W
Default:0x0000.0000



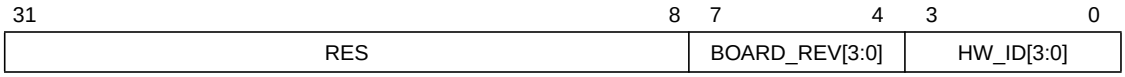
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

BOARD_REV[3:0]
Board Revision **[7:4]**
These bits could be used to enable interrupts for BOARD_REV[3:0] sampled from the multiplexed data lines at reset. However, software should never write to these bits.

HW_ID[3:0] **Hardware ID** **[3:0]**
These bits could be used to enable interrupts for HW_ID[3:0] sampled from the multiplexed data lines at reset. However, software should never write to these bits.

Register: 0x9000.000C

GPIO1_CLEAR
CPU:R/W
Default:0x0000.0000



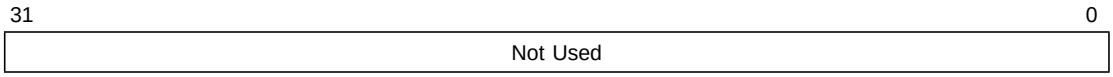
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

BOARD_REV[3:0]
Board Revision **[7:4]**
These bits could be used to disable interrupts for BOARD_REV[3:0] sampled from the multiplexed data lines at reset. However, software should never write to these bits.

HW_ID[3:0] **Hardware ID** **[3:0]**
These bits could be used to disable interrupts for HW_ID[3:0] sampled from the multiplexed data lines at reset. However, software should never write to these bits.

Register: 0x9000.0010

GPIO1_DDR
CPU:R/W
Default:0x0000.0000

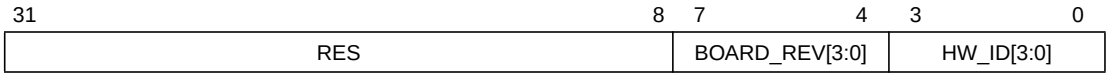


Not Used **[31:0]**

These bits are not used because the BOARD_REV[3:0] and HW_ID[3:0] bits in the GPIO1_DATA_IN register are always inputs.

Register: 0x9000.0014

GPIO1_DATA_OUT
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

BOARD_REV[3:0]
Board Revision **[7:4]**
These bits should never be used.

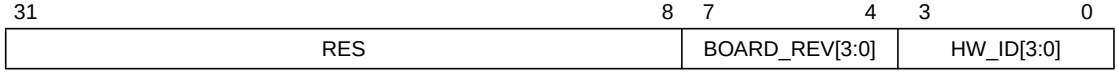
HW_ID[3:0] **Hardware ID** **[3:0]**
These bits should never be used.

Register: 0x9000.0018

GPIO1_DATA_IN

CPU:R

Default:Depends on Input Pull-Up or Pull-Down Resistors on the Board.



RES **Reserved** **[31:8]**

These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

BOARD_REV[3:0]

Board Revision **[7:4]**

These bits are latched in during reset according to the values set with pull-up or pull-down resistors on the board revision pins. The BOARD_REV[3:0] input pins are multiplexed with data lines DATA[83:80], respectively.

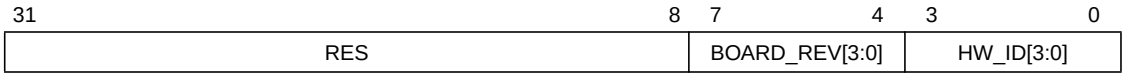
HW_ID[3:0] **Hardware ID** **[3:0]**

These bits are latched in during reset according to the values set with pull-up or pull-down resistors on the hardware ID pins. The HW_ID[3:0] input pins are multiplexed with data lines DATA[85, 84, 79, 78], respectively.

HW_ID[3:0]	Description
0b0000	24 10/100 Mbits/s ports plus 2 1000 Mbits/s ports
0b0001	12 10/100 Mbits/s ports plus 2 1000 Mbits/s ports

Register: 0x9000.001C

GPIO1_POL
CPU:R/W
Default:0x0000.0000



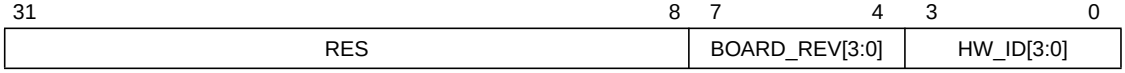
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

BOARD_REV[3:0]
Board Revision **[7:4]**
These bits should never be used.

HW_ID[3:0] **Hardware ID** **[3:0]**
These bits should never be used.

Register: 0x9000.0020

GPIO1_EDGE
CPU:R/W
Default:0x0000.0000



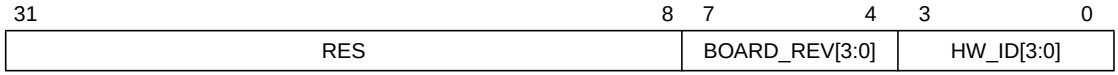
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

BOARD_REV[3:0]
Board Revision **[7:4]**
These bits should never be used.

HW_ID[3:0] **Hardware ID** **[3:0]**
These bits should never be used.

Register: 0x9000.0024

GPIO1_RESYNC
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

BOARD_REV[3:0]

Board Revision **[7:4]**
These bits should never be used.

HW_ID[3:0] **Hardware ID** **[3:0]**
These bits should never be used.

Register: 0x9000.0040

GPIO2_STAT
CPU:R
Default:0x0000.0000

31 … 8	7 6	5	4	3	2	1	0
RES	SP[1:0]	FF	CC	LM	DMP	DM	BM

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SP[1:0] **Spare** **[7:6]**
Not Used

FF **Fan Fault** **[5]**
This bit should never be used.

CC **Configure Clear** **[4]**
This bit should never be used.

LM **LED Mode** **[3]**
This bit indicates whether or not the LED Mode button is depressed (this bit is debounced in software).

LEDMOD	Description
0	Button depressed
1	Button not depressed

Note: This bit should never be used.

DMP **Dump** **[2]**
This bit reflects the state of the DUMP pin (G4), and indicates whether or not the DUMP jumper is installed. This bit is dedicated, not shared. This bit is debounced in software.

DMP	Description
0	Jumper installed (dump mode)
1	Jumper not installed (normal mode)

Note: This bit should never be used.

DM **Debug Mode** **[1]**

This bit indicates whether or not a DEBUG jumper is installed. It is sampled at reset from a mulitplexed data pin (A5). This bit is debounced in software.

DM	Description
0	Jumper installed (debug mode)
1	Jumper not installed (normal mode)

Note: This bit should never be used.

BM **Bench Mode** **[0]**

This bit indicates whether or not a BENCH MODE jumper is installed. It is sampled at reset from a mulitplexed data pin (B5). This bit is debounced in software.

BM	Description
0	Jumper installed (bench mode)
1	Jumper not installed (normal mode)

Note: This bit should never be used.

Register: 0x9000.0044

GPIO2_RAW_STAT
CPU:R/W
Default:0x0000.0000

31 … 8	7 6	5	4	3	2	1	0
RES	SP[1:0]	FF	CC	LM	DMP	DM	BM

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. These bits return 0 when read.

SP[1:0] **Spare** **[7:6]**
These are spare GPIO bits and have dedicated pins on the ASIC. SP[1] reflects the J4 pin and SP[0] reflects the K4 pin.

FF **Fan Fault** **[5]**
This bit directly reflects the state of the FAN_FAULT input pin (V23). When FAN_FAULT is asserted LOW, a fan fault condition exists. The FAN_FAULT input pin may also be driven from a temperature sensor circuit with an active LOW output.

CC **Configure Clear** **[4]**
This bit is connected directly to the CONFIG_CLEAR input switch pin (AC15). When the switch is depressed, this bit is 0. This input must be debounced in hardware.

LM **LED Mode** **[3]**
This bit indicates whether or not the LED Mode button is depressed (this bit should be debounced within the ASIC).

LEDMOD	Description
0	Button depressed
1	Button not depressed

DMP **Dump** **[2]**

This bit indicates whether or not the DUMP jumper is installed (this bit should be debounced within the ASIC).

DMP	Description
0	Jumper installed (dump mode)
1	Jumper not installed (normal mode)

DM **Debug Mode** **[1]**

This bit indicates whether or not a DEBUG jumper is installed. It is sampled at reset from a mulitplexed data pin (A5).

DM	Description
0	Jumper installed (debug mode)
1	Jumper not installed (normal mode)

BM **Bench Mode** **0**

This bit indicates whether or not a BENCH MODE jumper is installed. It is sampled at reset from a mulitplexed data pin (B5).

BM	Description
0	Jumper installed (bench mode)
1	Jumper not installed (normal mode)

Register: 0x9000.0048

GPIO2_EN_SET
CPU:R/W
Default:0x0000.0000

31	8	7	6	5	4	3	2	1	0
RES		SP[1:0]		FF	CC	LM	DMP	DM	BM

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. These bits return 0 when read.

SP[1:0] **Spare** **[7:6]**
These bits are used to enable interrupts for spare GPIO in case we find a need for them later.

FF **Fan Fault** **[5]**
This bit should never be enabled.

CC **Configure Clear** **[4]**
This bit should never be enabled.

LM **LED Mode** **[3]**
This bit should never be enabled.

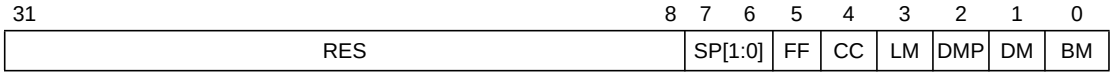
DMP **Dump** **[2]**
This bit should never be enabled.

DM **Debug Mode** **[1]**
This bit should never be enabled.

BM **Bench Mode** **[0]**
This bit should never be enabled.

Register: 0x9000.004C

GPIO2_CLEAR
CPU:W
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. These bits return 0 when read.

SP[1:0] **Spare** **[7:6]**
These bits are used to disable interrupts for spare GPIO.

FF **Fan Fault** **[5]**
This bit should never be used.

CC **Configure Clear** **[4]**
This bit should never be used.

LM **LED Mode** **[3]**
This bit should never be used.

DMP **Dump** **[2]**
This bit should never be used.

DM **Debug Mode** **[1]**
This bit should never be used.

BM **Bench Mode** **[0]**
This bit should never be used.

Register: 0x9000.0050

GPIO2_DDR
CPU:R/W
Default:0x0000.0000

31 … 8	7	6	5	4	3	2	1	0
RES	SP[1:0]		FF	CC	LM	DMP	DM	BM

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SP1 **Spare GPIO Bit 1 Direction** **7**
SP1 defines the direction for spare GPIO bit 1.

SPI	Description
0	Output
1	Input

SP0 **Spare GPIO Bit 0 Direction** **6**
SP0 defines the direction for spare GPIO bit 0.

SPI	Description
0	Output
1	Input

FF **Fan Fault GPIO Bit Direction** **5**
FF defines the direction for the FF bit. This bit must always be set as Input. Otherwise,the Fan Fault function is useless.

FF	Description
0	Output
1	Input

CC **Config Clear GPIO Bit Direction** **4**

CC defines the direction for the CC bit. This bit must always be set as Input. Otherwise, the Config Clear function is useless.

CC	Description
0	Output
1	Input

LM **LED Mode GPIO Bit Direction** **3**

LM defines the direction for the LM bit. This bit must always be set as Input. Otherwise, the LED Mode function is useless.

LM	Description
0	Output
1	Input

DMP **Dump GPIO Bit Direction** **2**

DMP defines the direction for the DMP bit. This bit must always be set as Input. Otherwise, the Dump function is useless.

DP	Description
0	Output
1	Input

DM **Debug Mode GPIO Bit Direction** **1**

DM defines the direction for the DM bit. This bit must always be set as Input. Otherwise, the Debug Mode function is useless.

DM	Description
0	Output
1	Input

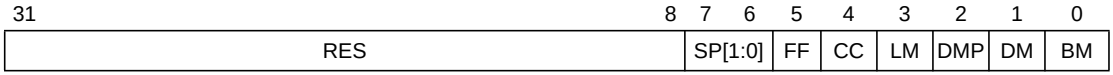
BM **Bench Mode GPIO Bit Direction** **0**

BM defines the direction for the BM bitThis bit must always be set as Input. Otherwise, the Bench Mode function is useless..

BM	Description
0	Output
1	Input

Register: 0x9000.0054

GPIO2_DATA_OUT
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SP[1:0] **Spare** **[7:6]**
These bits are spare GPIO.

FF **Fan Fault** **[5]**
This bit should never be used.

CC **Configure Clear** **[4]**
This bit should never be used.

LM **LED Mode** **[3]**
This bit should never be used.

DMP **Dump** **[2]**
This bit should never be used.

DM **Debug Mode** **[1]**
This bit should never be used.

BM **Bench Mode** **[0]**
This bit should never be used.

Register: 0x9000.0058

GPIO2_DATA_IN
CPU:R
Default:0x0000.0000

31 … 8	7	6	5	4	3	2	1	0
RES	SP[1:0]		FF	CC	LM	DMP	DM	BM

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SP1 **Spare GPIO Bit 1** **7**
SP1 reflects the state of spare GPIO bit 1.

SP0 **Spare GPIO Bit 0** **6**
SP0 reflects the state of spare GPIO bit 0.

FF **Fan Fault** **5**
This bit directly reflects the state of the FAN_FAULT input. When FAN_FAULT is asserted LOW, a fan fault condition exists. The FAN_FAULT input pin may also be driven from a temperature sensor circuit with an active LOW output.

FAN	Description
0	FAN_FAULT input is LOW (fault)
1	FAN_FAULT input is HIGH (no fault)

CC **Config Clear** **4**
This bit is connected directly to the Config Clear input switch. When the switch is depressed, the CC bit is 0. This input must be debounced in hardware.

CC	Description
0	Config Clear switch is depressed
1	Config Clear switch is not depressed

LM **LED Mode** **3**

This bit indicates whether or not the LED mode button is depressed. This bit is debounced and driven.

LM	Description
0	LED Mode button is depressed
1	LED Mode button is not depressed

DP **Dump** **2**

This bit indicates whether or not the DUMP jumper is installed. This input is debounced and driven.

DP	Description
0	DUMP jumper is installed
1	DUMP jumper is not installed

DM **Debug Mode** **1**

This bit indicates whether or not a DEBUG jumper is installed. The jumper is sampled at reset from a mulitplexed data pin (A5).

DM	Description
0	DEBUG jumper is installed (debug mode)
1	DEBUG jumper is not installed (normal mode)

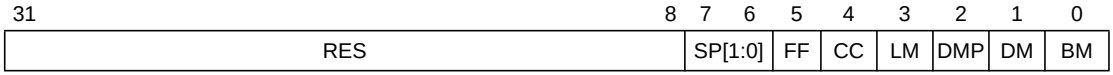
BM **Bench Mode** **0**

This bit indicates whether or not a BENCH jumper is installed. The jumper is sampled at reset from a mulitplexed data pin (B5).

BM	Description
0	BENCH jumper is installed (bench mode)
1	BENCH jumper is not installed (normal mode)

Register: 0x9000.005C

GPIO2_POL
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SP[1:0] **Spare** **[7:6]**
These bits are used to determine the polarity of the interrupts for spare GPIO.

FF **Fan Fault** **[5]**
This bit is used to determine the polarity of the interrupt for the FAN_FAULT input.

CC **Configure Clear** **[4]**
This bit is used to determine the polarity of the interrupt for the CONFIG_CLEAR input.

LM **LED Mode** **[3]**
This bit is used to determine the polarity of the interrupt for the LED Mode button.

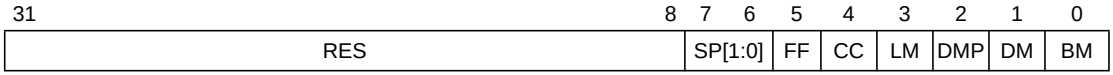
DMP **Dump** **[2]**
This bit is used to determine the polarity of the interrupt for the DUMP jumper.

DM **Debug Mode** **[1]**
This bit should never be enabled.

BM **Bench Mode** **[0]**
This bit should never be enabled.

Register: 0x9000.0060

GPIO2_EDGE
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SP[1:0] **Spare** **[7:6]**
These bits are used to determine if the interrupts for spare GPIO are edge or level based.

FF **Fan Fault** **[5]**
This bit is used to determine if the interrupt for the FAN_FAULT input is edge or level based.

CC **Configure Clear** **[4]**
This bit is used to determine if the interrupt for the CONFIG_CLEAR input is edge or level based.

LM **LED Mode** **[3]**
This bit is used to determine if the interrupt for the LED Mode button is edge or level based.

DMP **Dump** **[2]**
This bit is used to determine if the interrupt for the DUMP jumper is edge or level based.

DM **Debug Mode** **[1]**
This bit should never be enabled.

BM **Bench Mode** **[0]**
This bit should never be enabled.

Register: 0x9000.0064

GPIO2_RESYNC
CPU:R/W
Default:0x0000.0000

31 … 8	7 6	5	4	3	2	1	0
RES	SP[1:0]	FF	CC	LM	DMP	DM	BM

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SP[1:0] **Spare** **[7:6]**
These bits are used to determine if the interrupts for spare GPIO are resynchronized to the CPU clock.

FF **Fan Fault** **[5]**
This bit is used to determine if the interrupt for the FAN_FAULT input is resynchronized to the CPU clock.

CC **Configure Clear** **[4]**
This bit is used to determine if the interrupt for the CONFIG_CLEAR input is resynchronized to the CPU clock.

LM **LED Mode** **[3]**
This bit is used to determine if the interrupt for the LED Mode button is resynchronized to the CPU clock.

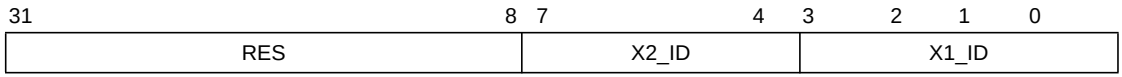
DMP **Dump** **[2]**
This bit is used to determine if the interrupt for the DUMP jumper is resynchronized to the CPU clock.

DM **Debug Mode** **[1]**
This bit should never be enabled.

BM **Bench Mode** **[0]**
This bit should never be enabled.

Register: 0x9000.0080

GPIO3_STAT
CPU:R
Default:0x0000.0000



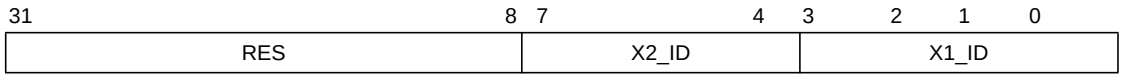
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

X2_ID **Transceiver 2 ID** **[7:4]**
These should never be used.

X1_ID **Transceiver 1 ID** **[3:0]**
These should never be used.

Register: 0x9000.0084

GPIO3_RAW_STAT
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

X2_ID **Transceiver 2 ID** **[7:4]**
These bits indicate the hardware ID of the Gigabit transceiver installed in Slot 2.

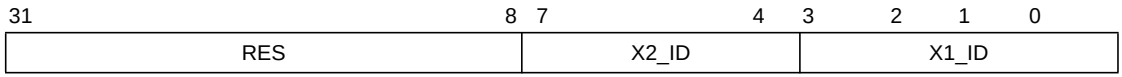
Note: 0b0111 = no transceiver installed

X1_ID **Transceiver 1 ID** **[3:0]**
These bits indicate the hardware ID of the Gigabit transceiver installed in Slot 1.

Note: 0b0111 = no transceiver installed

Register: 0x9000.0088

GPIO3_EN_SET
CPU:R/W
Default:0x0000.0000



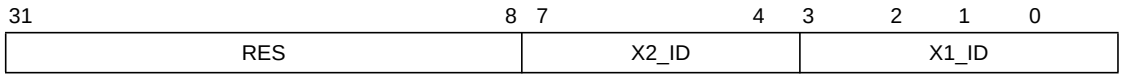
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

X2_ID **Transceiver 2 ID** **[7:4]**
These bits should never be set.

X1_ID **Transceiver 1 ID** **[3:0]**
These bits should never be set.

Register: 0x9000.008C

GPIO3_CLEAR
CPU:W
Default:0x0000.0000



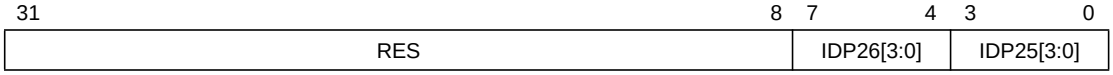
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

X2_ID **Transceiver 2 ID** **[7:4]**
These bits should never be set.

X1_ID **Transceiver 1 ID** **[3:0]**
These bits should never be set.

Register: 0x9000.0090

GPIO3_DDR
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

IDP26[3:0] **Direction of Port 26 Transceiver ID GPIO Bits** **[7:4]**
IDP26[3:0] define the direction for the X2_ID bits.

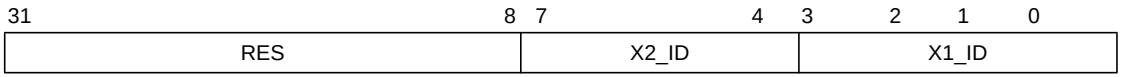
IDP26[n]	Description
0	Output
1	Input

IDP25[3:0] **Direction of Port 25 Transceiver ID GPIO Bits** **[3:0]**
IDP25[3:0] define the direction for the X1_ID bits.

IDP25[n]	Description
0	Output
1	Input

Register: 0x9000.0094

GPIO3_DATA_OUT
CPU:R/W
Default:0x0000.0000



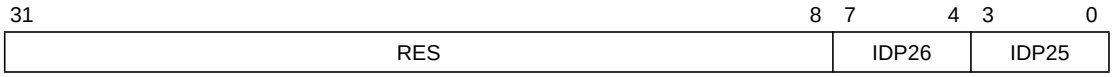
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

X2_ID **Transceiver 2 ID** **[7:4]**
These bits should never be written.

X1_ID **Transceiver 1 ID** **[3:0]**
These bits should never be written.

Register: 0x9000.0098

GPIO3_DATA_IN
CPU:R
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

IDP26[3:0] **Port 26 Transceiver ID** **[7:4]**
These bits indicate the hardware ID of the Gigabit transceiver installed in slot 2 (port 26).

IDP26[3:0]	Function	Interface
0b0000	TBI	1000BASE-LX Transceiver
0b0001	TBI	1000BASE-SX Transceiver
0b0010	MII	10/100BASE-TX Transceiver
0b0011	MII	100BASE-FX Transceiver
0b0100	MII/GMII	100/1000BASE-T Transceiver
0b0101	NA	1000BASE-LX Long Haul Transceiver
0b0110	TBI	1000BASE-CX Transceiver
0b0111	NA	No Transceiver Installed
0b1000	TBI	1000BASE-LX Transceiver - MTRJ
0b1001	TBI	Reserved
0b1010	NA	Reserved
0b1011	NA	Reserved
0b1100	NA	Reserved
0b1101	NA	Reserved
0b1110	NA	Reserved
0b1111	NA	Reserved

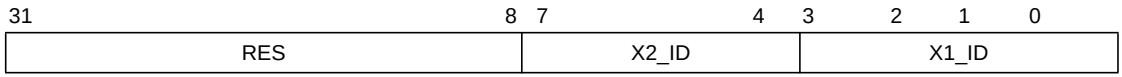
IDP25 **Port 25 Transceiver ID** **[3:0]**

These bits indicate the hardware ID of the Gigabit transceiver installed in slot 1 (port 25).

IDP26[3:0]	Function	Interface
0b0000	TBI	1000BASE-LX Transceiver
0b0001	TBI	1000BASE-SX Transceiver
0b0010	MII	10/100BASE-TX Transceiver
0b0011	MII	100BASE-FX Transceiver
0b0100	MII/GMII	100/1000BASE-T Transceiver
0b0101	NA	1000BASE-LX Long Haul Transceiver
0b0110	TBI	1000BASE-CX Transceiver
0b0111	NA	No Transceiver Installed
0b1000	TBI	1000BASE-LX Transceiver - MTRJ
0b1001	TBI	Reserved
0b1010	NA	Reserved
0b1011	NA	Reserved
0b1100	NA	Reserved
0b1101	NA	Reserved
0b1110	NA	Reserved
0b1111	NA	Reserved

Register: 0x9000.009C

GPIO3_POL
CPU:R/W
Default:0x0000.0000



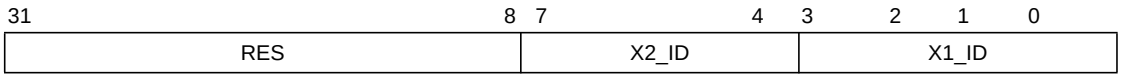
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

X2_ID **Transceiver 2 ID** **[7:4]**
These bits should never be written.

X1_ID **Transceiver 1 ID** **[3:0]**
These bits should never be written.

Register: 0x9000.00A0

GPIO3_EDGE
CPU:R/W
Default:0x0000.0000



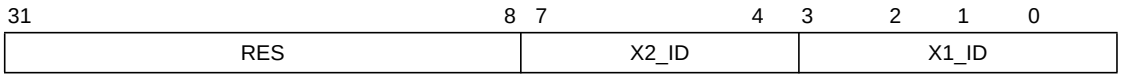
RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

X2_ID **Transceiver 2 ID** **[7:4]**
These bits should never be written.

X1_ID **Transceiver 1 ID** **[3:0]**
These bits should never be written.

Register: 0x9000.00A4

GPIO3_RESYNC
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

X2_ID **Transceiver 2 ID** **[7:4]**
These bits should never be written.

X1_ID **Transceiver 1 ID** **[3:0]**
These bits should never be written.

Register: 0x9000.00C0

GPIO4_STAT
CPU:R
Default:0x0000.0000

31 … 8	7 6	5	4	3	2 1	0
RES	SS[1:0]	LP26	LP25	UMEN	UMPR[1:0]	RMON

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SS **System Speed** **[7:6]**
These pins are multiplexed with SDRAM_DATA[67:66] and are sampled and latched at reset. The state of the pins determines the device operating clock frequency. The system uses the state of the pins to adjust logic such as the Memory Controller.

SS	Description
0b00	System Clock is 125 MHz
0b01	System Clock is 133 MHz
0b10	System Clock is 143 MHz
0b11	System Clock is 150 MHz (default)

LP26 **Port 26 Link Status/Signal Detect** **5**
LP26 reflects the port 26 link status and optical power detect status, as indicated from the LINK/SIGDET signal of the port 26 transceiver.

SS	Description
0	No signal detected (default)
1	Signal detected

LP25 **Port 25 Link Status/Signal Detect** **4**

LP25 reflects the port 25 link status and optical power detect status, as indicated from the LINK/SIGDET signal of the port 25 transceiver.

SS	Description
0	No signal detected (default)
1	Signal detected

UMEN **Unknown Multicast Enable** **3**

UMEM	Description
0	Unknown multicast addresses are forwarded to the CPU. The UMPR[1:0] field determines to which of the four priority queues the packet is queued (default).
1	Unknown multicast addresses are not forwarded to the CPU and the UMPR[1:0] field is ignored.

UMPR[1:0] **Unknown Multicast Priority** **[2:1]**

This field determines which of the four CPU queues is used for unknown multicast addresses. This field is only used when the UMEN bit is 1.

UMPR[1:0]	Description
0b00	CPU queue 0 (default)
0b01	CPU queue 1
0b10	CPU queue 2
0b11	CPU queue 3

RMON **RMON Enable** **0**

Setting this bit starts the RMON sampling hardware on all ports. When this bit changes from 0 to 1, new random skip counts are loaded from NEXT_SKIPCOUNT into SKIPCOUNT for each port. If software does not want RMON samples from a given port, it should set the SAMPLE_MASK and SAMPLE_BASE values for the port to very large numbers, then discard the samples for the port when the CPU receives them.

Register: 0x9000.00C4

GPIO4_RAW_STAT
CPU:R/W
Default:0x0000.0000

31 … 8	7 6	5	4	3	2 1	0
RES	SS[1:0]	LP26	LP25	UMEN	UMPR[1:0]	RMON

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SS **System Speed** **[7:6]**
These pins are multiplexed with SDRAM_DATA[67:66] and are sampled and latched at reset. The state of the pins determines the device operating clock frequency. The system uses the state of the pins to adjust logic such as the Memory Controller.

SS	Description
0b00	System Clock is 125 MHz
0b01	System Clock is 133 MHz
0b10	System Clock is 143 MHz
0b11	System Clock is 150 MHz (default)

LP26 **Port 26 Link Status/Signal Detect** **5**
LP26 reflects the port 26 link status and optical power detect status, as indicated from the LINK/SIGDET signal of the port 26 transceiver.

SS	Description
0	No signal detected (default)
1	Signal detected

LP25 **Port 25 Link Status/Signal Detect** **4**

LP26 reflects the port 25 link status and optical power detect status, as indicated from the LINK/SIGDET signal of the port 25 transceiver.

SS	Description
0	No signal detected (default)
1	Signal detected

UMEN **Unknown Multicast Enable** **3**

UMEM	Description
0	Unknown multicast addresses are forwarded to the CPU. The UMPR[1:0] field determines to which of the four priority queues the packet is queued (default).
1	Unknown multicast addresses are not forwarded to the CPU and the UMPR[1:0] field is ignored.

UMPR[1:0] **Unknown Multicast Priority** **[2:1]**

This field determines which of the four CPU queues is used for unknown multicast addresses. This field is only used when the UMEN bit is 1.

UMPR[1:0]	Description
0b00	CPU queue 0 (default)
0b01	CPU queue 1
0b10	CPU queue 2
0b11	CPU queue 3

RMON **RMON Enable** **0**

Setting this bit starts the RMON sampling hardware on all ports. When this bit changes from 0 to 1, new random skip counts are loaded from NEXT_SKIPCOUNT into SKIPCOUNT for each port. If software does not want RMON samples from a given port, it should set the SAMPLE_MASK and SAMPLE_BASE values for the port to very large numbers, then discard the samples for the port when the CPU receives them.

Register: 0x9000.00C8

GPIO4_EN_SET
CPU:R/W
Default:0x0000.0000

31 — 8	7	6	5	4	3	2 — 1	0
RES	SS		LP26	LP25	UMEN	UMPR[1:0]	RMON

When a 1 is written to a bit in the GPIO4_EN_SET register, an interrupt is enabled for the corresponding GPIO4 signal, based on the settings in the GPIO4_POL and GPIO4_EDGE registers. Writing a 0 has no effect. To clear a bit, use the GPIO4_CLEAR register.

RES — **Reserved** — **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SS — **System Speed** — **[7:6]**
These bits should always be written as 0.

LP26 — **Port 26 Link Status/Signal Detect** — **5**
When a 1 is written to this bit, an interrupt is enabled for the port 26 Link Status/Signal Detect, based on the settings in the GPIO4_POL and GPIO4_EDGE registers. Writing a 0 has no effect.

LP25 — **Port 25 Link Status/Signal Detect** — **4**
When a 1 is written to this bit, an interrupt is enabled for the port 25 Link Status/Signal Detect, based on the settings in the GPIO4_POL and GPIO4_EDGE registers. Writing a 0 has no effect.

UMEN — **Unknown Multicast Enable** — **3**
This bit should always be written as 0.

UMPR[1:0] — **Unknown Multicast Priority** — **[2:1]**
These bits should always be written as 0.

RMON — **RMON Enable** — **0**
This bit should always be written as 0.

Register: 0x9000.00CC

GPIO4_CLEAR
CPU:W
Default:0x0000.0000

31 — 8	7	6	5	4	3	2 — 1	0
RES	SS	LP26	LP25	UMEN	UMPR[1:0]	RMON	

When a 1 is written to a bit in the GPIO4_CLEAR register, the corresponding bit is cleared in the GPIO4_EN_SET register. Writing a 0 has no effect.

Register: 0x9000.00D0

GPIO4_DDR
CPU:R/W
Default:0x0000.0000

31 — 8	7 — 6	5	4	3	2 — 1	0
RES	SS	LP26	LP25	UMEN	UMPR[1:0]	RMON

This register is used to set the direction for the corresponding signal. When a bit is 1, the corresponding GPIO signal is an input, When a bit is 0, the corresponding GPIO signal is an output.

RES — **Reserved** — **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SS — **System Speed** — **[7:6]**
These bits should always be written as 0.

LP26 — **Port 26 Link Status/Signal Detect** — **5**
This bit should always be written as 1.

LP25 — **Port 25 Link Status/Signal Detect** — **4**
This bit should always be written as 1.

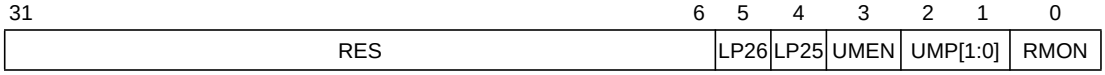
UMEN — **Unknown Multicast Enable** — **3**
This bit should always be written as 0.

UMPR[1:0] — **Unknown Multicast Priority** — **[2:1]**
These bits should always be written as 0.

RMON — **RMON Enable** — **0**
This bit should always be written as 0.

Register: 0x9000.00D4

GPIO4_DATA_OUT
CPU:R/W
Default:0x0000.00x0



This register is used to write the data value for the corresponding GPIO bit.

RES **Reserved** **[31:6]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

LP26 **Port 26 Link Status/Signal Detect** **5**
This bit is an input only and should not be written.

LP25 **Port 25 Link Status/Signal Detect** **4**
This bit is an input only and should not be written.

UMEN **Unknown Multicast Enable** **3**

UMEN	Description
0	Unknown multicast addresses are forwarded to the CPU. The UMP[1:0] field determines to which of the four priority queues the packet is queued (default).
1	Unknown multicast addresses are not forwarded to the CPU and the UMP[1:0] field is ignored.

UMP[1:0] **Unknown Multicast Priority** **[2:1]**
This field determines which of the four CPU queues is used for unknown multicast addresses. This field is only used when the UMEN bit is 0.

UMPR[1:0]	Description
0b00	CPU queue 0 (default)
0b01	CPU queue 1
0b10	CPU queue 2
0b11	CPU queue 3

RMON	**RMON Enable**	**0**
	Setting this bit starts the RMON sampling hardware on all ports. When this bit changes from 0 to 1, new random skip counts are loaded from NEXT_SKIPCOUNT into SKIPCOUNT for each port. If software does not want RMON samples from a given port, it should set the SAMPLE_MASK and SAMPLE_BASE values for the port to very large numbers, then discard the samples for the port when the CPU receives them.	

Register: 0x9000.00D8

GPIO4_DATA_IN
CPU:R
Default:0x0000.0000

31 — 8	7 — 6	5	4	3 — 0
RES	SS[1:0]	LS26	LS25	NU

This register is used to read inputs.

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SS **System Speed** **[7:6]**
These pins are multiplexed with SDRAM_DATA[67:66] and are sampled and latched at reset. The state of the pins determines the device operating clock frequency. The system uses the state of the pins to adjust logic such as the Memory Controller.

SS	Description
0b00	System Clock is 125 MHz (default)
0b01	System Clock is 133 MHz
0b10	System Clock is 143 MHz
0b11	System Clock is 150 MHz

LP26 **Port 26 Link Status/Signal Detect** **5**
LP26 reflects the port 26 link status and optical power detect status, as indicated from the LINK/SIGDET signal of the port 26 transceiver.

SS	Description
0	No signal detected (default)
1	Signal detected

LP25 **Port 25 Link Status/Signal Detect** **4**

LP25 reflects the port 25 link status and optical power detect status, as indicated from the LINK/SIGDET signal of the port 25 transceiver.

SS	Description
0	No signal detected (default)
1	Signal detected

NU **Not Used** **[3:0]**

Register: 0x9000.00DC

GPIO4_POL
CPU:R
Default:0x0000.0000

31 … 8	7 6	5	4	3	2 1	0
RES	SS	LS26	LS25	UMEN	UMP	RMON

The GPIO4_POL register determines whether a GPIO interrupt is activated from a corresponding bit that is a 1 (HIGH level) or transitions from 0 to 1 (rising edge), or from a bit that is a 0 (LOW level) or transitions from 1 to 0 (falling edge).

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SS **System Speed** **[7:6]**
Not applicable

LP26 **Port 26 Link Status/Signal Detect** **5**

LP26	Description
0	LOW Level/Falling Edge (default)
1	HIGH Level/Rising Edge

LP25 **Port 25 Link Status/Signal Detect** **4**

LP25	Description
0	LOW Level/Falling Edge (default)
1	HIGH Level/Rising Edge

UMEN **Unknown Multicast Enable** **3**
Not applicable

UMP **Unknown Multicast Priority** **[2:1]**
Not applicable

RMON **RMON Enable** **0**
Not applicable

Register: 0x9000.00E0

GPIO4_EDGE
CPU:R/W
Default:0x0000 00FF

31 8	7 6	5	4	3	2 1	0
RES	SS	LS26	LS25	UMEN	UMP	RMON

The GPIO4_EDGE register determines whether a GPIO interrupt is activated from a corresponding bit that set at a level or from a corresponding bit that experiences a transition (edge).

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SS **System Speed** **[7:6]**

SS	Description
0	Level triggered (default)
1	Edge triggered

LP26 **Port 26 Link Status/Signal Detect** **5**

LP26	Description
0	Level triggered
1	Edge triggered (default)

LP25 **Port 25 Link Status/Signal Detect** **4**

LP25	Description
0	Level triggered
1	Edge triggered (default)

UMEN **UM Enable Unknown Multicast** **3**
Not applicable

UMP **UM Priority Unknown Multicast** **[2:1]**
Not applicable

RMON **RMON Enable** **0**
Not applicable

Register: 0x9000.00E4

GPIO4_RESYNC
CPU:R/W
Default:0x0000 00FF

31 … 8	7	6	5	4	3	2	1	0
RES	SS		LS26	LS25	UMEN	UMP		RMON

The GPIO4_RESYNC register determines whether the corresponding GPIO signal is resynchronized to the internal clock when the signal is configured as an input.

RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. The bits return 0 when read.

SS **System Speed** **[7:6]**

SS	Description
0	No resync (default)
1	Resync

LP26 **Port 26 Link Status/Signal Detect** **5**

LP26	Description
0	No resync
1	Resync (default)

LP25 **Port 25 Link Status/Signal Detect** **4**

LP25	Description
0	No resync
1	Resync (default)

UMEN **Unknown Multicast Enable** **3**

UMEN	Description
0	No resync
1	Resync (default)

UMP **Unknown Multicast Priority** **[2:1]**

UMP	Description
0	No resync
1	Resync (default)

RMON **RMON Enable** **0**

RMON	Description
0	No resync
1	Resync (default)

8.3 Interrupt Registers

This section lists the registers that are used to set, clear, and read the status of the L64324 interrupts.

Register: 0x8000.0000/0x8000.0100

IRQ_STAT/FIQ_STAT
CPU:R
Default:0x0xxx.xxxx

31 27	26	25	24	23	22 17	16	15	14	13	12	11	10	9	8	7	6	5 4	3 0
RES	AL	I26	I25	I24	TIM[5:0]	WT	PI	R	26U	25U	24U	SW	TX	RX	CTT	CRR	UI[1:0]	GP[4:1]

RES — **Reserved** **[31:27]**
These bits are reserved.

AL — **Unaligned Access Interrupt Status** **26**
This bit reflects the interrupt that occurs when the CPU attempts to perform an unaligned access to memory.

I26 — **INTR_26 Latched Interrupt Status** **25**
This bit reflects the active LOW interrupt input driven from the transceiver module for port 26.

I25 — **INTR_25 Latched Interrupt Status** **24**
This bit reflects the active LOW interrupt input driven from the transceiver module for port 25.

I24 — **INTR_1_24 All PHYs Interrupt Latched** **23**
This bit reflects the OR'd condition of all the active LOW PHY interrupts for ports 1 through 24.

TIM[5:0] — **Timer Trigger Interrupts** **[22:17]**
These bits are the six comparator outputs from the timer.

WT — **Watchdog Timer Interrupt** **16**
This bit is set when the Watchdog Timer expires.

PI — **Programmed Interrupt** **15**
This bit is set/cleared on writes to the FIQ/IRQ register.

R — **Reserved** **14**
This bit is reserved. Software writes the bit as 0 and reads it as don't care.

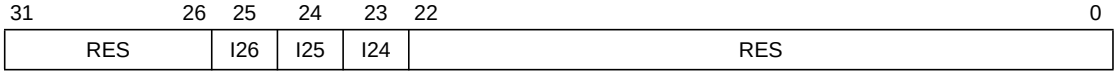
26U — **INTR_26 Unlatched Interrupt Status** **13**
This bit reflects the active LOW interrupt input driven from the transceiver module for port 26.

25U	**INTR_25 Unlatched Interrupt Status** This bit reflects the active LOW interrupt input driven from the transceiver module for port 25.	**12**
24U	**INTR_1_24 All PHYs Interrupt Unlatched** This bit reflects the OR'd condition of all the active LOW PHY interrupts for ports 1 through 24.	**11**
SW	**Switch** The switching core sets this bit to notify the CPU that an event has take place that the CPU needs to deal with. See LAN_INT_EN in the Packet Tx/Rx section.	**10**
TX	**CPU PKT TX Interrupt** The switch network interface sets this bit when a packet from the CPU has been removed from the CPU transmit ring.	**9**
RX	**CPU PKT RX Interrupt** The switch network interface sets this bit when a packet destined for the CPU Memory is received.	**8**
CTT	**CT Communication TX** This is the transmit empty flag from the ARM7. It is cleared when the external device attached to the JTAG port shifts out the contents of the transmit buffer in the ARM7 module.	**7**
CRR	**CR Communication RX** This is transmit full flag from the ARM7. It is set when the external device attached to the JTAG port shifts in data to be received by the ARM7.	**6**
UI[1:0]	**UART Interrupts** The UART sets these bits when an interrupt condition occurs. There is a single interrupt bit per UART.	**[5:4]**
GP[4:1]	**GP4–GP1 Interrupts** These bits indicate if an interrupt has occurred because of activity on the corresponding GPIO pins.	**[3:0]**

Register: 0x8000.0004/0x8000.0104

IRQ_RAW_STAT/FIQ_RAW_STAT
CPU:R/W
Default:0x0x0.0000

The IRTQ_RAW_STAT/FIQ_RAW_STAT registers contain the premasked interrupts on read. On writes, the register is used to clear active interrupts. A 1 appearing in a bit position indicates that an interrupt is active. Write a 1 to each bit location to be cleared.



RES **Reserved** **[31:26]**
These bits are reserved.

I26 **INTR_26** **25**
Writing a 1 to this bit clears the latched interrupt caused from the active LOW interrupt input driven from the transceiver module for port 26.

I25 **INTR_25** **24**
Writing a 1 to this bit clears the latched interrupt caused from the active LOW interrupt input driven from the transceiver module for port 25.

I24 **INTR_1_24** **23**
Writing a 1 to this bit clears the latched interrupt caused from the OR'd condition of all the active LOW PHY interrupts for ports 1 through 24.

RES **Reserved** **[22:0]**
These bits are reserved.

Register: 0x8000.008/0x8000.0108

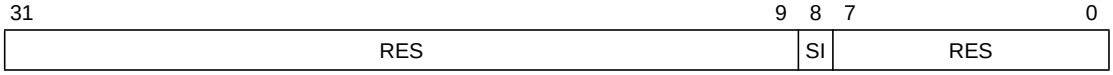
IRQ_EN_SET/FIQ_EN_SET
CPU:R/W
Default:0x0000.0000

31	0
INT_MASK[31:0]	

INT_MASK **Interrupt Mask** **[31:0]**
Writing a 1 masks the corresponding interrupt.

Register: 0x8000.0010

FIQ_IRQ_SWI
CPU:R/W
Default:0x0000.0000



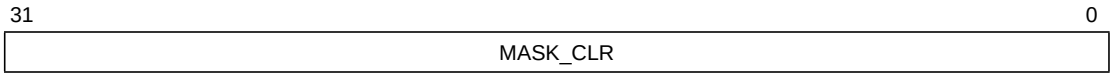
RES **Reserved** **[31:9]**
These bits are reserved.

SI **Software Interrupt** **8**
Writing a 1 to this bit position generates a software interrupt.

RES **Reserved** **[7:0]**
These bits are reserved.

Register: 0x8000.000C/0x8000.010C

IRQ_CLEAR/FIQ_CLEAR
CPU:R/W
Default:0x0000.0000



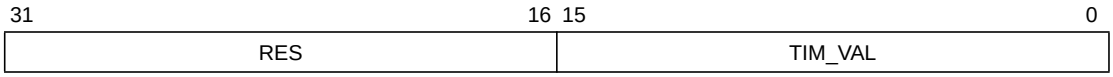
MASK_CLR **Clear Interrupt Mask** **[31:0]**
Writing a 1 clears the corresponding interrupt mask.

8.4 TIMER Registers

The Timer registers configure the operation of the two on-chip timer units and six trigger point registers. The base address of the Timer registers is 0x8400.0000.

Register: 0x8400.0000

TIMER_1
CPU:R/W
Default:0x0000.xxxx

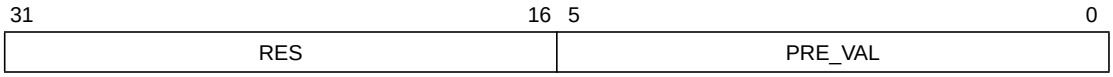


RES **Reserved** **[31:16]**
These bits are reserved.

TIM_VAL **Timer Value** **[15:0]**
This field shows the current value of the timer. If the FRZ bit is not set in the TIMER_1_CNTL register, this field shows different values at different times, depending on the current count value.

Register: 0x8400.0004

TIMER_1_PSCALE
CPU:R/W
Default:0x0000.xxxx



RES **Reserved** **[31:16]**
These bits are reserved.

PRE_VAL **Prescale Value** **[15:0]**
The timer prescale register is used to prescale the timer input clock, which decrements the timer. PRE_VAL forms the time base for the timer. Writing a 0 to PRE_VAL allows the timer to run at the same rate as the input clock to the timer block.

Register: 0x8400.0008

TIMER_1_CNTL
CPU:R/W
Default:0x0000.000xx

31 … 5	4	3	2 … 0
RES	FRZ	UP	TRIG_SEL[2:0]

RES **Reserved** **[31:5]**
These bits are reserved.

FRZ **Freeze Bit Controls the Timer** **4**
When this bit is set to 0, Timer 1 is in the run mode. When FRZ is 1, the timer freezes at the current timer value.

UP **Count Direction** **3**
When this bit is set, the timer counts up; otherwise it counts down.

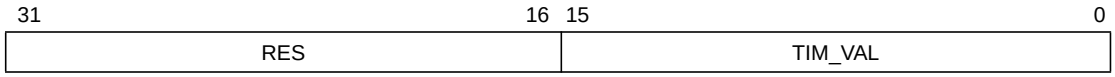
TRIG_SEL[2:0]
Trigger Select **[2:0]**
The TRIG_SEL[2:0] field selects whether the timer operates as an interval or periodic timer. If it is set up as a periodic timer (TRIG SEL field is 0b001 to 0b110), a Trigger point register is selected that causes a reload of the timer when a comparison occurs between the Timer and the selected Trigger Point register. When this match occurs, the Timer is reloaded with 0 and continues to count. If the timer is set up as an interval timer (TRIG_SEL[2:0] field is 0b000), the timer counts to its maximum or minimum value and stops.

TRIG_SEL[2:0]	Description
0b000	Interval timer
0b001	Trigger Point 1 register selected

TRIG_SEL[2:0]	Description
0b010	Trigger Point 2 register selected
0b011	Trigger Point 3 register selected
0b100	Trigger Point 4 register selected
0b101	Trigger Point 5 register selected
0b110	Trigger Point 6 register selected
0b111	Reserved

Register: 0x8400.000C

TIMER_2
CPU:R/W
Default:0x0000.00xx

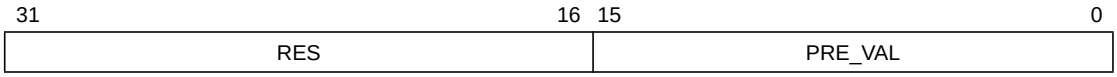


RES **Reserved** **[31:16]**
These bits are reserved.

TIM_VAL **Timer Value** **[15:0]**
This field shows the current value of timer. If the FRZ bit is not set in the TIMER_2_CNTL register, this field shows different values at different times, depending on the current count value.

Register: 0x8400.0010

TIMER_2_PSCALE
CPU:R/W
Default:0x0000.00xx



RES **Reserved** **[31:16]**
These bits are reserved.

PRE_VAL **Prescale Value** **[15:0]**
The timer prescale register is used to prescale the timer input clock, which decrements the timer. PRE_VAL forms the time base for the timer. Writing a 0 to PRE_VAL allows the timer to run at the same rate as the input clock to the timer block.

Register: 0x8400.0014

TIMER_2_CNTL
CPU:R/W
Default:0x0000.000x

31 … 5	4	3	2 … 0
RES	FRZ	UP	TRIG_SEL[2:0]

RES **Reserved** **[31:5]**
These bits are reserved.

FRZ **Freeze Bit Controls the Timer** **4**
When this bit is cleared, Timer 2is in the run mode. When FRZ is 1, the timer freezes at the current timer value.

UP **Count Direction** **3**
When this bit is set, the timer counts up; otherwise it counts down.

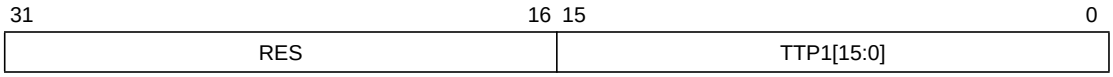
TRIG_SEL[2:0]
Trigger Select **[2:0]**
The TRIG_SEL[2:0] field selects whether the timer operates as an interval or periodic timer. If it is set up as a periodic timer (TRIG SEL field is 0b001 to 0b110), a Trigger point register is selected that causes a reload of the timer when a comparison occurs between the Timer and the selected Trigger Point register. When this match occurs, the Timer is reloaded with 0 and continues to count. If the timer is set up as an interval timer (TRIG_SEL[2:0] field is 0b000), the timer counts to its maximum or minimum value and stops.

TRIG_SEL[2:0]	Description
0b000	Interval timer
0b001	Trigger Point 1 register selected

TRIG_SEL[2:0]	Description
0b010	Trigger Point 2 register selected
0b011	Trigger Point 3 register selected
0b100	Trigger Point 4 register selected
0b101	Trigger Point 5 register selected
0b110	Trigger Point 6 register selected
0b111	Reserved

Register: 0x8400.0018

TIMER_TRG_1
CPU:R/W
Default:0x0000.xxxx

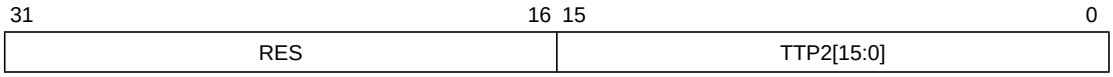


RES **Reserved** **[31:16]**
These bits are reserved.

TTP1[15:0] **Timer Trigger Point 1** **[15:0]**
The TTP1[15:0] value defines the point at which an interrupt is generated. An interrupt is generated when the chosen timer value matches the trigger point value.

Register: 0x8400.001C

TIMER_TRG_2
CPU:R/W
Default:0x0000.xxxx

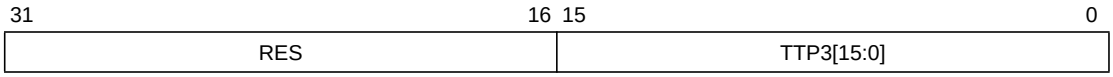


RES **Reserved** **[31:16]**
These bits are reserved.

TTP2[15:0] **Timer Trigger Point 2** **[15:0]**
The TTP2[15:0] value defines the point at which an interrupt is generated. An interrupt is generated when the chosen timer value matches the trigger point value.

Register: 0x8400.0020

TIMER_TRG_3
CPU:R/W
Default:0x0000.xxxx

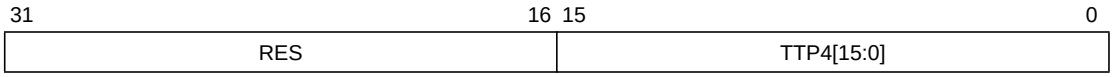


RES **Reserved** **[31:16]**
These bits are reserved.

TTP3[15:0] **Timer Trigger Point 3** **[15:0]**
The TTP3[15:0] value defines the point at which an interrupt is generated. An interrupt is generated when the chosen timer value matches the trigger point value.

Register: 0x8400.0024

TIMER_TRG_4
CPU:R/W
Default:0x0000.xxxx

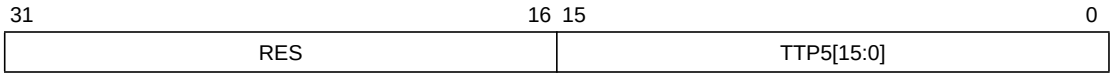


RES **Reserved** **[31:16]**
These bits are reserved.

TTP4[15:0] **Timer Trigger Point 4** **[15:0]**
The TTP4[15:0] value defines the point at which an interrupt is generated. An interrupt is generated when the chosen timer value matches the trigger point value.

Register: 0x8400.0028

TIMER_TRG_5
CPU:R/W
Default:0x0000.xxxx

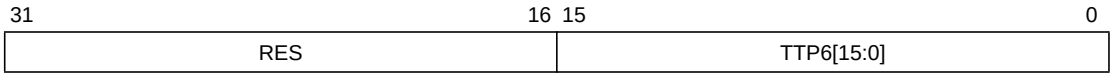


RES **Reserved** **[31:16]**
These bits are reserved.

TTP5[15:0] **Timer Trigger Point 5** **[15:0]**
The TTP5[15:0] value defines the point at which an interrupt is generated. An interrupt is generated when the chosen timer value matches the trigger point value.

Register: 0x8400.002C

TIMER_TRG_6
CPU:R/W
Default:0x0000.xxxx



RES **Reserved** **[31:16]**
These bits are reserved.

TTP6[15:0] **Timer Trigger Point 6** **[15:0]**
The TTP6[15:0] value defines the point at which an interrupt is generated. An interrupt is generated when the chosen timer value matches the trigger point value.

Register: 0x8400.0030

TIMER_TRG_CNTL
CPU:R/W
Default:0x0000.00xx

31 ... 6	5	4	3	2	1	0
RES	TPS6	TPS5	TPS4	TPS3	TPS2	TPS1

RES **Reserved** **[31:6]**
These bits are reserved.

TPS6 **Trigger Point Select for Trigger Point Register 6** **5**
The TPS6 bit defines which time base (Timer 1 or Timer 2) Trigger Point register 6 is compared against to generate an interrupt.

TPS6	Description
0	Compared to Timer 1
1	Compared to Timer 2

TPS5 **Trigger Point Select for Trigger Point Register 5** **4**
The TPS5 bit defines which time base (Timer 1 or Timer 2) Trigger Point register 5 is compared against to generate an interrupt.

TPS5	Description
0	Compared to Timer 1
1	Compared to Timer 2

TPS4 **Trigger Point Select for Trigger Point Register 4** **3**
The TPS4 bit defines which time base (Timer 1 or Timer 2) Trigger Point register 4 is compared against to generate an interrupt.

TPS4	Description
0	Compared to Timer 1
1	Compared to Timer 2

TPS3 **Trigger Point Select for Trigger Point Register 3** **2**

The TPS3 bit defines which time base (Timer 1 or Timer 2) Trigger Point register 3 is compared against to generate an interrupt.

TPS3	Description
0	Compared to Timer 1
1	Compared to Timer 2

TPS2 **Trigger Point Select for Trigger Point Register 2** **1**

The TPS2 bit defines which time base (Timer 1 or Timer 2) Trigger Point register 2 is compared against to generate an interrupt.

TPS2	Description
0	Compared to Timer 1
1	Compared to Timer 2

TPS1 **Trigger Point Select for Trigger Point Register 1** **0**

The TPS1 bit defines which time base (Timer 1 or Timer 2) Trigger Point register 1 is compared against to generate an interrupt.

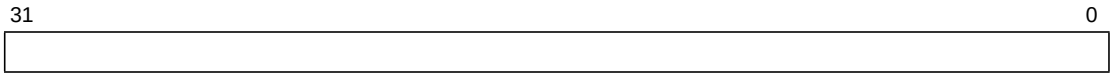
TPS1	Description
0	Compared to Timer 1
1	Compared to Timer 2

8.5 Remap and Pause Registers

The Remap and Pause registers have a base address of 0x8800.0000.

Register: 0x8800.0000

REMAP_REG
CPU:R/W
Default:0xxxxx.xxxx

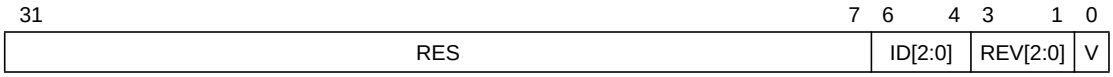


Writing any value to this register causes the processor to enter a pause state and wait for an interrupt.

Note: To write to this register, an access control mechanism is required. The user must read the access control lock register, located at 0x9800.0000, then invert bits[7:0] of the register and write them back to 0x9800.0004. The REMAP_REG register may then be written.

Register: 0x8800.0010

REMAP_REV
CPU:R
Default:0x0000 0001



RES **Reserved** **[31:7]**
These bits are reserved.

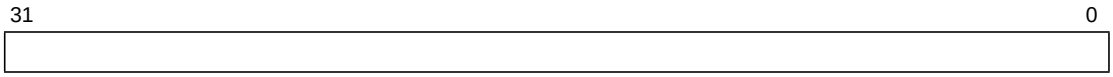
ID[2:0] **Chip Identification** **[6:4]**
This can be set to the revision number of the product and can be used to encode useful information. The L64324 Chip ID is 0b000 (24-port 10/100, 2-port Gigabit).

REV[2:0] **Chip Revision** **[3:1]**
This can be set to indicate the chip revision. The default is 0b000.

V **Valid** **0**
If this bit is set to 1, it indicates that additional information is available. The additional information will be specified later.

Register: 0x8800.0020

REMAP_REMAP
CPU:W
Default:0xxxxx.xxxx



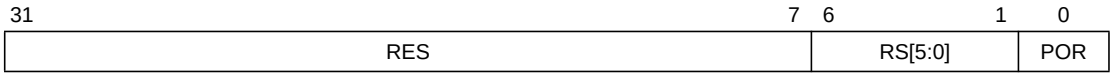
Writing any value to this register clears the reset memory map and makes the internal memory available.

Writing the REMAP_REMAP register causes the system memory map to change from the FLASH being located at address 0x0000.0000 (the reset value) to the DRAM being located at address 0x0000.0000.

Register: 0x8800.0030

REMAP_RST_STAT
CPU:R/W
Default:0x0000.0001

This register identifies the type of reset that has occurred. The register has a dual mechanism for setting and clearing bits, allowing independent bits to be altered with no knowledge of the other bits in the register. Reading this register returns the values of the reset status bits. Writing a 1 to a bit position causes that bit to be set. Writing a zero has no effect.



RES **Reserved** **[31:7]**
These bits are reserved.

RS[5:0] **Reset Status** **[6:1]**
These six bits keep track of the reset status or states that have been entered or completed after power-up. Software can use this register to record the completion of various reset or initialization events which have occurred and make this information available to other start-up or initialization processes.

On read, this register provides the current reset status. On writes, this location is used to set Reset Status Flags. When writing to this register each data bit which is HIGH causes the corresponding bit in the Reset Status register to be set. Data bits that are LOW have no effect on the corresponding bit in the Reset Status register.

RS[5:0]	Name	Description
5	SW	Reserved
4	Bench	Software sets this bit before it initiates a Bench Reset 0 = No Bench Reset has occurred 1 = Bench Reset has occurred
3	Console	Software sets this bit before it initiates a Console Reset 0 = No Console Reset has occurred 1 = Console Reset has occurred

RS[5:0]	Name	Description
2	Warm	Software sets this bit before it initiates a Warm Reset 0 = No Warm Reset has occurred 1 = Warm Reset has occurred
1	Cold	Software sets this bit before it initiates a Cold Reset 0 = No Cold Reset has occurred 1 = Cold Reset has occurred
0	Hard	Software sets this bit before it initiates a Hard Reset 0 = No Hard Reset has occurred 1 = Hard Reset has occurred

POR **Power on Reset** **0**

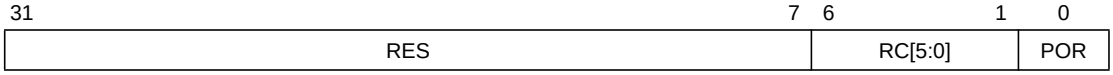
This bit reflects whether or not the box has gone through a power on reset (POR).

POR	Description
0	No POR since the flag was last cleared
1	POR has occurred

Note: To write to this register, an access control mechanism is required. The user must read the access control lock register, located at 0x9800.0000, then invert bits[7:0] of the register and write them back to 0x9800.0004. The REMAP_RST_STAT register may then be written.

Register: 0x8800.0034

REMAP_RST_CLR
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:7]**
These bits are reserved.

RC[5:0] **Reset Clear** **[6:1]**
These bits, when written as 1, clear the corresponding reset bits (RS[5:0]). This register is used along with Reset Status Set.

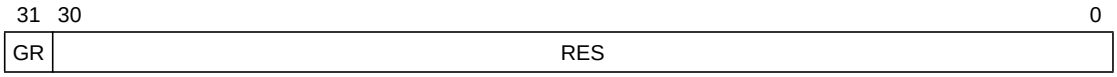
RC[5:0]	Name	Description
5	SW	Writing a 1 clears the SW bit in the REMAP_RST_STAT register
4	Bench	Writing a 1 clears the Bench bit in the REMAP_RST_STAT register
3	Console	Writing a 1 clears the Console bit in the REMAP_RST_STAT register
2	Warm	Writing a 1 clears the Warm bit in the REMAP_RST_STAT register
1	Cold	Writing a 1 clears the Cold bit in the REMAP_RST_STAT register
0	Hard	Writing a 1 clears the Hard bit in the REMAP_RST_STAT register

POR **Clear POR** **0**
This bit, when set clears the POR bit.

Note: To write to this register, an access control mechanism is required. The user must read the access control lock register, located at 0x9800.0000, then invert bits[7:0] of the register and write them back to 0x9800.0004. The REMAP_RST_CLR register may then be written.

Register: 0x8800.0038

REMAP_SRSR
CPU:R/W
Default:0x0000.0000



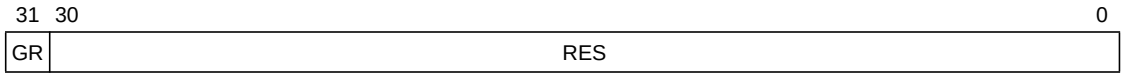
GR **Global Reset** **31**
When this bit is set, the entire chip is reset.

Note: To write to this register, an access control mechanism is required. The user must read the access control lock register, located at 0x9800.0000, then invert bits[7:0] of the register and write them back to 0x9800.0004. The REMAP_SRSR register may then be written.

RES **Reserved** **[30:0]**
These bits are reserved.

Register: 0x8800.003C

REMAP_SRCR
CPU:W
Default:0x0000.0000



GR **Global Reset** **31**
When this bit is set, the GR bit in the REMAP_SRSR register is cleared.

Note: To write to this register, an access control mechanism is required. The user must read the access control lock register, located at 0x9800.0000, then invert bits[7:0] of the register and write them back to 0x9800.0004. The REMAP_SRCR register may then be written.

RES **Reserved** **[30:0]**
These bits are reserved.

Register: 0x8800.0040

REMAP_HCR
CPU:R/W
Default:0x0000.000x

31	2	1	0
RES		CS7	DA

RES **Reserved** **[31:2]**

These bits are reserved.

CS7 **Chip Select 7 MAP** **1**

This bit controls the functionality of the Chip Select 7 (CS7n) signal from the SRAM/FLASH memory controller. This bit only affects the operation of CS7n after Remap has been cleared, previous to being cleared, CS7n is the only chip select that can be activated.

CS7	Description
0	CS7N is asserted only when address being accessed matches that of the CSN7 base register.
1	CSN7 is asserted both when the address being accessed matches that of the CSN7 base register and CSN0 base register.

DA **DRAM AVAIL** **0**

This bit is used by the decoder to select the mapping of the ARM memory region from 0x0000.0000 to 0x1000.0000.

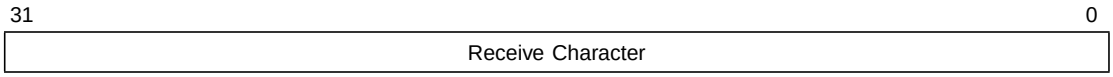
CS7	Description
0	SRAM/FLASH regions mapped in the range 0x1000.0000 to 0x1FFF_FFFF are also mapped in the range 0x0000.0000 to 0x0FFF.FFFF
1	SDRAM available

8.6 UART Registers

The UART registers have a base address of 0x8C00.0000.

Register: 0x8C00.0000

UART_RHR (Receive Holding Register)
CPU:R
Default:0xxxxx.xxxx



When the Receive Data Ready bit (RDR) is set in the UART_LSR register, these bits contain a character received over the serial channel.

This register is accessible when the baud rate counter latch enable bit (DLAB) in the UART_LCR register is 0.

Register: 0x8C00.0000

UART_THR (Transmit Holding Register)
CPU:W
Default:0xxxxx.xxxx

31	0
Transmit Character	

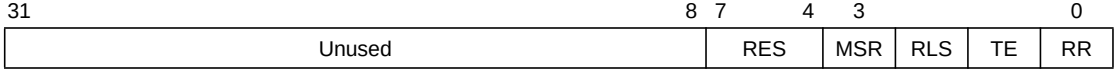
When the transmit holding empty bit (THE) is set in the UART_LSR register, the data written to the UART_THR register is transmitted out the serial port.

This register is accessible when the baud rate counter latch enable bit (DLAB) in the UART_LCR register is 0.

Register: 0x8C00.0004

UART_IER (Interrupt Enable Register)
CPU:R/W
Default:0x0000.000x

The UART_IER register masks incoming interrupts from receiver ready, transmitter empty, line status and modem status registers to the INT output pin.



RES **Reserved** **[7:4]**

These bits are reserved.

MSR **Modem Status Register Interrupt Enable/Disable** **3**

MSR	Description
0	Disable the modem status register interrupt.
1	Enable the modem status register interrupt.

RLS **Receive Line Status Interrupt Enable/Disable** **2**

RLS	Description
0	Disable the receiver line status interrupt.
1	Enable the receiver line status interrupt.

TE **Transmit Empty Interrupt Enable/Disable** **1**

TE	Description
0	Disable the transmitter empty interrupt.
1	Enable the transmitter empty interrupt.

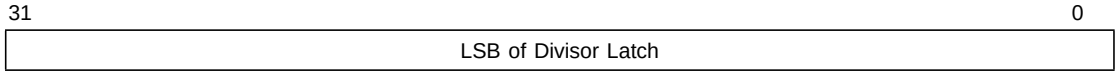
RR **Receiver Ready Interrupt Enable/Disable** **0**

RR	Description
0	Disable the receiver ready interrupt.
1	Enable the receiver ready interrupt.

Register: 0x8C00.0000

UART_LDR (LSB Divisor Latch Register)
CPU:W
Default:0xxxxx.xxxx

The UART contains a programmable Baud Rate Generator that is capable of tracking any clock input from DC to 8 MHz and dividing it by any divisor from 1 to 2^16 – 1. The output frequency of the Baudout is equal to 16x the transmission baud rate (Baudout = 16 x Baud Rate). Custom Baud Rates can be achieved by selecting proper divisor values for the MSB and LSB of the baud rate generator.

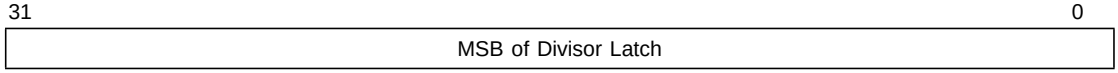


This register is accessible when the baud rate counter latch enable bit (DLAB) in the UART_LCR register is 1.

Register: 0x8C00.0004

UART_MDLR (MSB Divisor Latch Register)
CPU:W
Default:0xxxxx.xxxx

The UART contains a programmable Baud Rate Generator that is capable of tracking any clock input from DC to 8 MHz and dividing it by any divisor from 1 to 2^16 – 1. The output frequency of the Baudout is equal to 16x the transmission baud rate (Baudout = 16 x Baud Rate). Custom Baud Rates can be achieved by selecting proper divisor values for the MSB and LSB of baud rate generator.

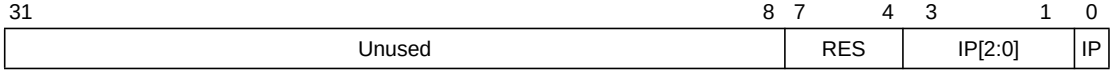


This register is accessible when the baud rate counter latch enable bit (DLAB) in the UART_LCR register is 1.

Register: 0x8C00.0008

UART_ISR (Interrupt Status Register)
CPU:R
Default:0x0000.000x

The UART provides four-level prioritized interrupt conditions to minimize software overhead during data character transfers. The UART_ISR register provides the source of the interrupt in a prioritized manner. During the read cycle, the UART provides the highest interrupt level to be serviced by the CPU. No other interrupts are acknowledged until the particular interrupt is serviced.



Unused **[31:8]**
These bits are not used.

RES **Reserved** **[7:4]**
These bits are reserved.

IP[2:0] **Interrupt Priority** **[3:1]**
The logical combination of these bits provides the highest priority interrupt pending.

IP[2:0]	Description
0x000	MSR (Modem Status Register)
0x001	TXRDY (Transmitter Holding Register Empty)
0x010	RXRDY (Received Data Ready)
0x011	LSR (Receiver Line Status Register)
0x110	RXRDY (Received Data Timeout)

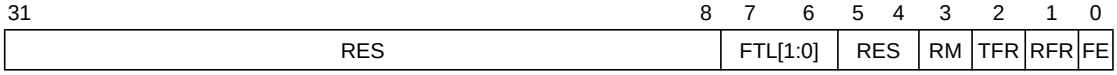
IP **Interrupt Pending** **0**
This bit indicates whether or not an interrupt is pending.

IP	Description
0	An interrupt is pending and the ISR contents may be used as a pointer to the appropriate interrupt service routine.
1	No interrupt pending

Register: 0x8C00.0008

UART_FCR (FIFO Control Register)
CPU:W
Default:0x0000.00xx

This register enables the FIFOs, clears the FIFOs, sets the receiver FIFO trigger level, and selects the type of DMA signaling.



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. These bits return 0 when read.

FIFO_TL[1:0] **Trigger Level** **[7:6]**
These bits are used to set the trigger level for the receiver FIFO interrupt.

FIFO_TL[1:0]	FIFO Trigger Level
0b00	01
0b01	04
0b10	08
0b11	14

RES **Reserved** **[5:4]**
These bits are reserved.

RM **Rx/Tx Ready Mode** **3**

RM	Description
0b0	Mode 0 used for RXRDY and TXRDY pins
0b1	Mode 1 used for RXRDY and TXRDY pins

Transmit operation in Mode 0:

When the 16550 is in 16450 mode (FCR bit 0 = 0) or in the FIFO mode (FCR bit 0 = 1 and FCR bit 3 = 0), when there are no characters in the transmit FIFO or transmit holding register, the TXRDYn pin is asserted LOW. After the pin has been asserted, it is deasserted (HIGH) after the first character is loaded into the transmit holding register.

Receive operation in Mode 0:

When the 16550 is in 16450 mode (FCR bit 0 = 0) or in the FIFO mode (FCR bit 0 = 1 and FCR bit 3 = 0) and there is at least one character in the receive FIFO, the RXRDYn pin is asserted LOW. After the pin has been asserted, the pin is deasserted (HIGH) when there are no more characters in the receiver FIFO.

Transmit operation in Mode 1:

When the 16550 is in the FIFO mode (FCR bit 0 = 1 and FCR bit 3 = 1), the TXRDYn pin is deasserted (HIGH) when the transmit FIFO is completely full. It is asserted (LOW) when one or more FIFO locations are empty.

Receive operation in Mode 1:

When the 16550 is in the FIFO mode (FCR bit 0 = 1 and FCR bit 3 = 1) and the trigger level or the timeout has been reached, the RXRDYn is asserted (LOW). After it is asserted, it is deasserted when there are no more characters in the FIFO.

TFR **Tx FIFO Reset** **2**

TFR	Description
0	No change.
1	Clears the contents of the transmit FIFO and resets its counter logic to 0 (the transmit shift register is not cleared or altered). This bit returns to 0 after clearing the FIFOs.

RFR **Rx FIFO Reset** **1**

RFR	Description
0	No change.
1	Clears the contents of the receive FIFO and resets its counter logic to 0 (the transmit shift register is not cleared or altered). This bit returns to 0 after clearing the FIFOs.

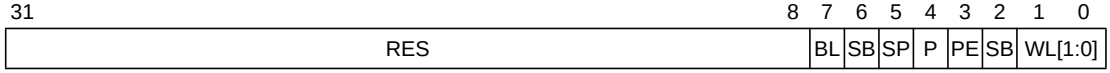
FE **FIFO Enable** **0**

RFR	Description
0	Disable the transmit and receive FIFO
1	Enable the transmit and receive FIFO. This bit should be enabled before setting the FIFO trigger levels

Register: 0x8C00.000C

UART_LCR (Line Control Register)
CPU:W
Default:0x0000.00xx

The line Control Register specifies the asynchronous data communication format. The word length, stop bits, and parity can be selected through this register.



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. These bits return 0 when read.

BL **Baud Latch** **7**
The internal baud rate counter latch enable (DLAB).

BL	Description
0	disabled, normal operation
1	enabled

SB **Set Break** **6**
SB is the Break control bit. When set, it causes a break condition to be transmitted (the Tx output is forced LOW).

SP **Set Parity** **5**
This bit is used to force the parity bit to 1 or 0.

SP	Description
0	Parity bit is 0 if P is 1
1	Parity bit is 1 if P is 0

P **Parity** **4**
If the PE bit is 1 (enabled), P selects even or odd parity format.

P	Description
0	Parity is odd
1	Parity is even

Odd parity means that the character always has an odd number of ones. Even parity means the character always has an even number of ones.

PE **Parity Enable** **3**

PE	Description
0	Parity is not generated or checked
1	Parity is generated on transmission and checked on receive

SB **Stop Bits** **2**

The SB bit specifies the number of stop bits.

SB	Description
0	1 Stop bit
1	2 Stop bits

WL[1:0] **Word Length** **[1:0]**

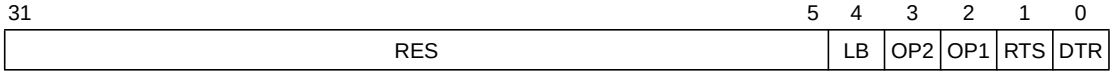
These two bits specify the word length to be transmitted or received.

WL[1:0]	Word Length
0b00	9
0b10	7
0b11	8

Register: 0x8C00.0010

UART_MCR (Modem Control Register)
CPU:W
Default:0x0000.000x

This register controls the interface with an attached modem or a peripheral device (RS232).



RES **Reserved** **[31:5]**

These bits are reserved.

LB **Loopback** **4**

During Loopback mode, the transmitter output (TX) is set HIGH (Mark condition), and the receiver input (RX), CTS, DSR, CD, and RI are disabled. Internally, the transmitter output is connected to the receiver input and the DTR, RTS, OP1, and OP2 signals are connected to the modem control inputs. In this mode, the receiver and transmitter interrupts are fully operational, but the interrupt sources are now the lower four bits of the Modem Control Register instead of the four Modem Control inputs. The IER still controls the interrupts.

LB	Description
0	Normal operating mode
1	Enable local loopback mode (diagnostic)

OP2 **Option 2** **3**

On IBM compatible computers, the OP2 bit controls the board's interrupt 3-state buffer.

OP2	Description
0	Option 2 output is HIGH
1	Option 2 output is LOW

OP1 **OP1** **2**

OP1	Description
0	Option 1 output is HIGH
1	Option 1 output is LOW

RTS **RTS** **1**

RTS	Description
0	Force RTS (Request to Send) HIGH
1	Force RTS LOW

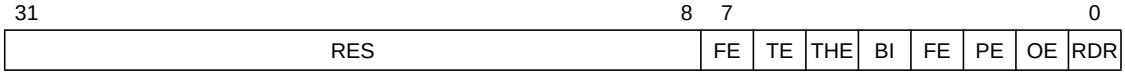
DTR **DTR** **0**

DTR	Description
0	Force DTR (Data Terminal Ready) HIGH
1	Force DTR LOW

Register: 0x8C00.0014

UART_LSR (Line Status Register)
CPU:R
Default:0x0000.00xx

This register provides the status of the data transfer to the CPU.



RES **Reserved** **[31:8]**

These bits are reserved. Software should write 0 to these bits. These bits return 0 when read.

FE **FIFO Error** **7**

FE	Description
0	Normal
1	At least one parity error, framing error or break indicator is in the FIFO. FE is cleared when the register is read.

TE **Tx Empty** **6**

TE	Description
0	Transmitter holding and shift registers are full.
1	Transmit holding register is empty. In FIFO mode, this bit is set to 1 whenever the transmitter FIFO and transmit shift register are empty.

THE **Tx Holding Empty** **5**

THE	Description
0	The transmit holding register is full. The 16550 UART will not accept any data for transmission.
1	The transmitter holding register (or FIFO) is empty, and the CPU can load the next character.

BI **Break Interrupt** **4**

BI	Description
0	No break condition (normal)
1	Receiver received a break signal (RX was LOW for one character time frame).

FE **Framing Error** **3**

FE	Description
0	No framing error (normal)
1	Framing error received. Received data did not have a valid stop bit.

PE **Parity Error** **2**

PE	Description
0	No parity error (normal)
1	Parity error. Receive data does not have correct parity information.

OE **Overrun Error** **1**

OE	Description
0	No overrun error (normal)
1	Overrun error. A character arrived before the receive holding register was emptied or, if FIFOs are enabled, an overrun error will occur only after the FIFO is full and the next character has been completely received in the shift register. Note that character in the shift register is overwritten, but it is not transferred to the FIFO.

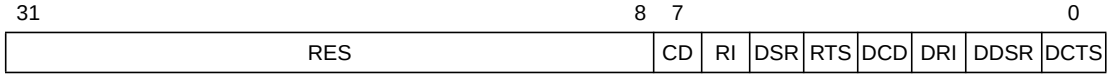
RDR **Rx Data Ready** **0**

RDR	Description
0	No data in receive holding register or FIFO
1	Data has been received and saved in the receive holding register or FIFO.

Register: 0x8C00.0018

UART_MSR (Modem Status Register)
CPU:R/W
Default:0x0000.00xx

This register provides the current state of the control lines from the modem or peripheral to the CPU. Four bits of this register indicate the changed information. These bits are set whenever a control input from the modem changes state. They are cleared whenever the CPU reads the register.



RES **Reserved** **[31:8]**
These bits are reserved. Software should write 0 to these bits. These bits return 0 when read.

CD **CD** **7**
This bit is equivalent to the OP2 bit in the UART_MCR register during local loopback mode. It is the complement to the CD input.

RI **RI** **6**
This bit is equivalent to the OP1 bit in the UART_MCR register during local loopback mode. It is the complement of the RI input.

DSR **DSR** **5**
This bit is equivalent to DTR in the UART_MCR register during local loopback mode. It is the complement of the CTS input.

RTS **RTS** **4**
When asserted, this bit indicates that RTS changed state.

DCD **Delta CD** **3**
When asserted, this bit indicates that CD changed state.

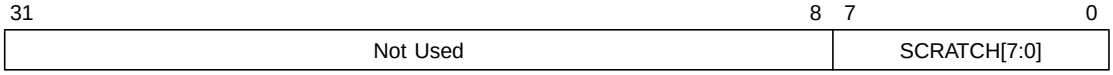
DRI **Delta RI** **2**
When asserted, this bit indicates that RI changed state.

DDSR	**Delta DSR** When asserted, this bit indicates that DSR changed state.	**1**
DCTS	**Delta CTS** When asserted, this bit indicates that CTS changed state.	**0**

Register: 0x8C00.001C

UART_SRn (Scratch Register)
CPU:R/W
Default:0xxxxx.xxxx

Eight bits of general-purpose scratch information can be stored in this register.



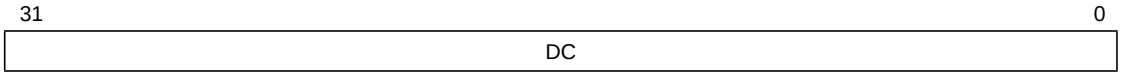
Register: 0x8C00.0020

UART_SSR (Speed Sense Register)
CPU:R/W
Default:0x0000.xxxx

To use this register the software needs to:

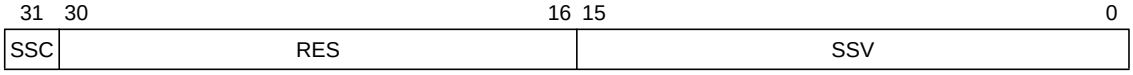
1. Write any value to the Speed Sense Register. This clears the Speed Sense Complete (SSC) bit as well as the Speed Sense Value (SSV) field within the UART_SSR register.
2. Poll the UART_SSR register until the SSC bit is 1 (active).
3. Use the resulting Speed Sense Value (SSV) field to update the baud rate generator (Divisor Latch) in the UART.

Write Register:



DC **Don't care** **[31:0]**
A write of any value to this register begins the Speed Sense process.

Read Register:



SSC **Speed Sense Complete** **31**
When SSC is 1, it indicates that a speed sense operation is complete and the speed sense value is valid. This bit is cleared on read.

RES **Reserved** **30:16**
These bits are reserved.

SSV[15:0] **Speed Sense Value** **15:0**
SSV[15:0] is the count of the length of the first start bit.

8.7 10/100 Mbits/s Port Registers

Each of the ports in the L64324 ASIC can operate in the 10/100 Mbits/s mode. Ports 1 through 24, which operate at 10/100 Mbits/s, are located in the E110 core. Ports 25 and 26 are Gigabit ports that can be configured to operate at 10/100 Mbits/s. Each port has a collection of registers detailed in the following pages. The register operation is the same for each port, and the base address of each register is listed in the table below.

Table 8.2 10/100 Mbits/s Port Base Addresses

Port Number	Base Address
1	0x3400.0000
2	0x3401.0000
3	0x3402.0000
4	0x3403.0000
5	0x3404.0000
6	0x3405.0000
7	0x3406.0000
8	0x3407.0000
9	0x3408.0000
10	0x3409.0000
11	0x340A.0000
12	0x340B.0000
13	0x340C.0000
14	0x340D.0000
15	0x340E.0000
16	0x340F.0000
17	0x3410.0000
18	0x3411.0000
19	0x3412.0000

Table 8.2 10/100 Mbits/s Port Base Addresses (Cont.)

Port Number	Base Address
20	0x3413.0000
21	0x3414.0000
22	0x3415.0000
23	0x3416.0000
24	0x3417.0000
25	0x3418.0000 (if 10/100 Mbits/s operation is enabled)
26	0x3419.0000 (if 10/100 Mbits/s operation is enabled)

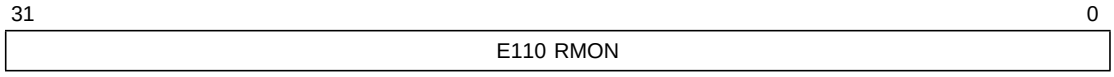
Register: 0x340X.0000

RMON_TOTAL_COUNT
CPU:R/W
Default:0x0000.0000

E110 RMON_TOTAL_COUNT (etherStatsOctets) [31:0]

Register: 0x340X.0004

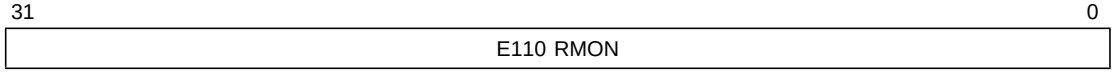
RMON_LENGTH64
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsPkts64Octets) [31:0]

Register: 0x340X.0008

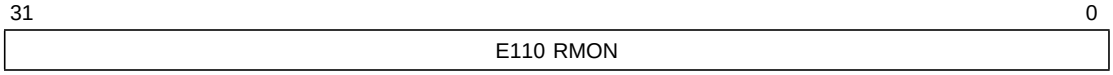
RMON_LENGTH65
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsPkts65to127Octets) [31:0]

Register: 0x340X.000C

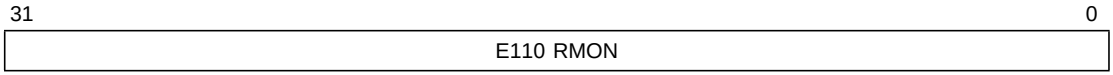
RMON_LENGTH128
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsPkts128to255Octets) [31:0]

Register: 0x340X.0010

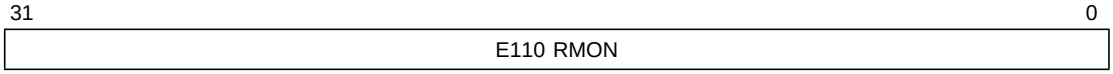
RMON_LENGTH256
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsPkts256to511Octets) [31:0]

Register: 0x340X.0014

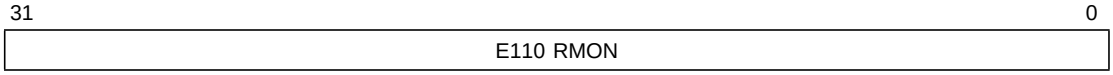
RMON_LENGTH512
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsPkts512to1023Octets) [31:0]

Register: 0x340X.0018

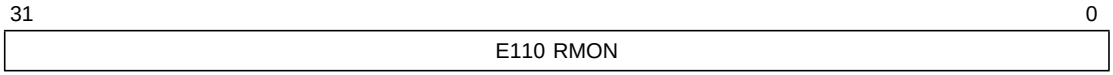
RMON_LENGTH1024
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsPkts1024to1518Octets) [31:0]

Register: 0x340X.001C

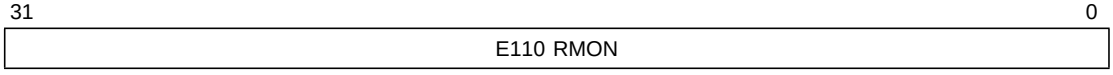
RMON_LENGTH1519
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsPkts1519to15220Octets) [31:0]

Register: 0x340X.0020

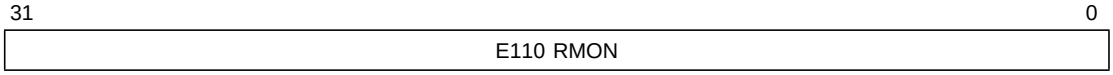
RMON_SYMBOL_ERROR
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3StatsSymbolErrors) [31:0]

Register: 0x340X.0024

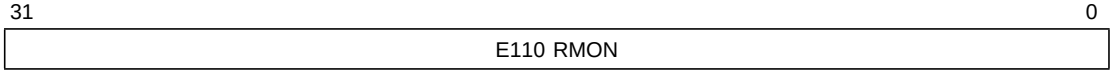
RMON_CRC_ERROR
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3StatsFCSErrors) [31:0]

Register: 0x340X.0028

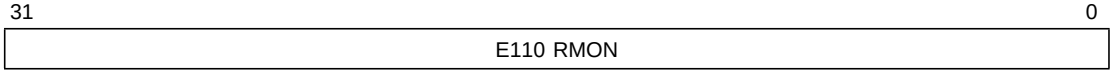
RMON_UNDERSIZE
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsUndersizePkts) [31:0]

Register: 0x340X.002C

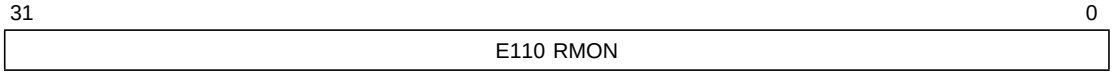
RMON_OVERSIZE
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsOversizePkts) [31:0]

Register: 0x340X.0030

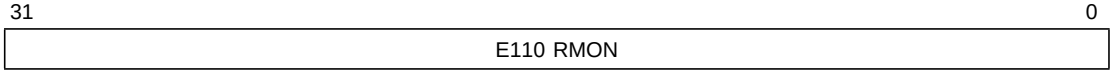
RMON_JABBER
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsJabbers) [31:0]

Register: 0x340X.0034

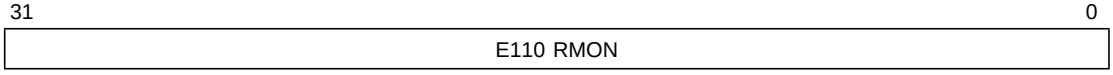
RMON_FRAGMENT
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsFragments) [31:0]

Register: 0x340X.0038

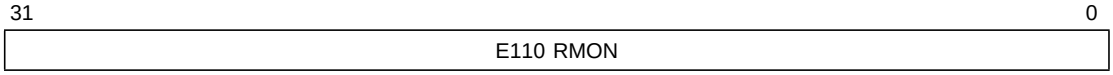
RMON_RX_COUNT
CPU:R/W
Default:0x0000.0000



E110 RMON (ifInOctets) [31:0]

Register: 0x340X.003C

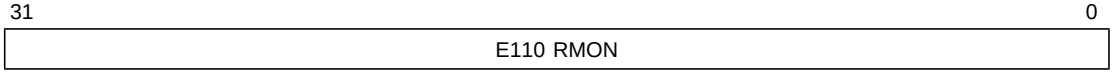
RMON_RX_BRDCAST
CPU:R/W
Default:0x0000.0000



E110 RMON (ifInBroadicastPkts) [31:0]

Register: 0x340X.0040

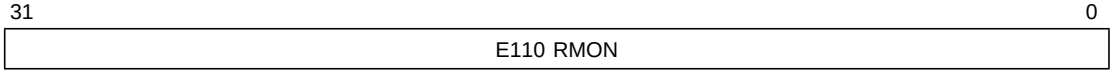
RMON_RX_MLTCAST
CPU:R/W
Default:0x0000.0000



E110 RMON (ifInMulticastPkts) [31:0]

Register: 0x340X.0044

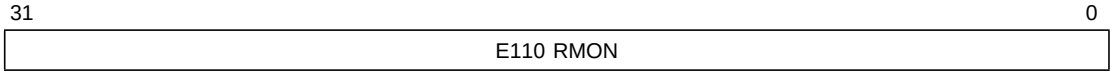
RMON_RX_UNICAST
CPU:R/W
Default:0x0000.0000



E110 RMON (ifInUcastPkts) [31:0]

Register: 0x340X.0048

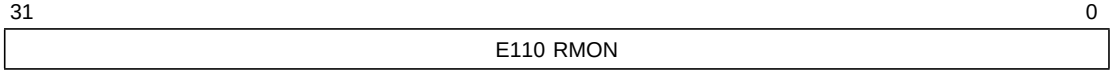
RMON_RX_PAUSE
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3PauseFrames) [31:0]

Register: 0x340X.004C

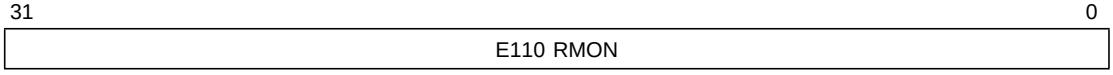
RMON_RX_UNKNOWN
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3ControlInUnknownOpcodes) [31:0]

Register: 0x340X.0050

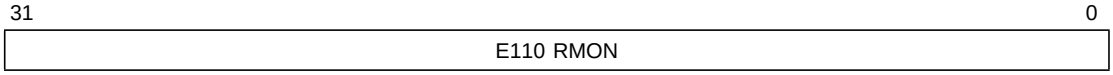
RMON_RX_OVERFLOW
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3InternalMacReceiveErrors) [31:0]

Register: 0x340X.0054

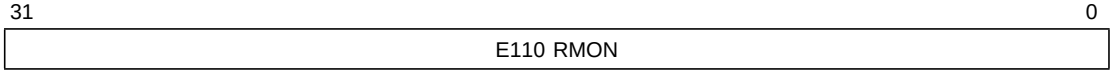
RMON_FALSE_CARRIER
CPU:R/W
Default:0x0000.0000



E110 RMON (ifMauFalseCarriers) [31:0]

Register: 0x340X.0058

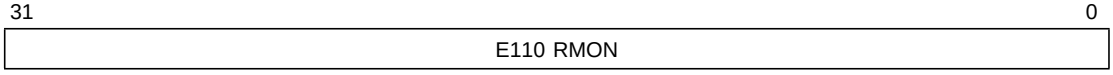
RMON_RX_PKT_DROP
CPU:R/W
Default:0x0000.0000



E110 RMON (RXDroppedPackets) [31:0]

Register: 0x340X.005C

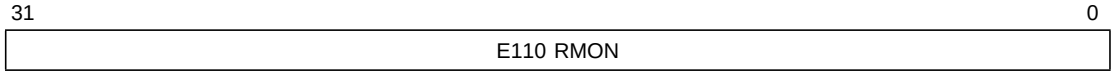
RMON_TX_COUNT
CPU:R/W
Default:0x0000.0000



E110 RMON (ifOutOctets) [31:0]

Register: 0x340X.0060

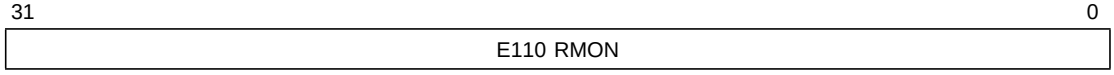
RMON_TX_UNDER_RUN
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3StatsInternalMacTransmitError) [31:0]

Register: 0x340X.0064

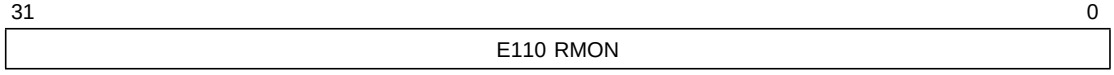
RMON_TX_BRDCAST
CPU:R/W
Default:0x0000.0000



E110 RMON (ifOutBroadcastPkts) [31:0]

Register: 0x340X.0068

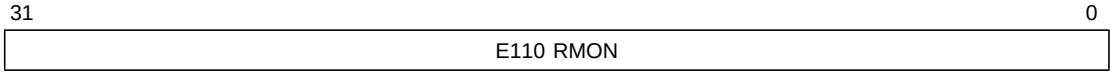
RMON_TX_MLTCAST
CPU:R/W
Default:0x0000.0000



E110 RMON (ifOutMulticastPkts) [31:0]

Register: 0x340X.006C

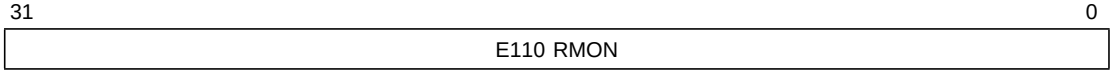
RMON_TX_UNICAST
CPU:R/W
Default:0x0000.0000



E110 RMON (ifOutUnicastPkts) [31:0]

Register: 0x340X.0070

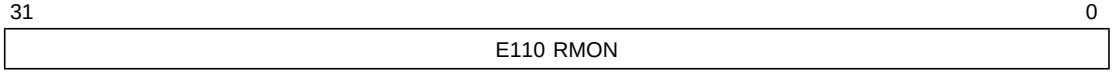
RMON_TX_PAUSE
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3OutPauseFrames) [31:0]

Register: 0x340X.0074

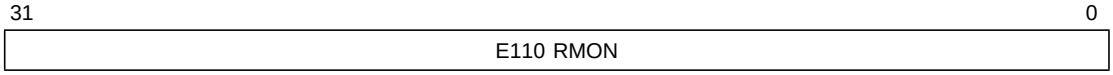
RMON_DRIBBLE_ERR
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3StatsAlignmentErrors) [31:0]

Register: 0x340X.0078

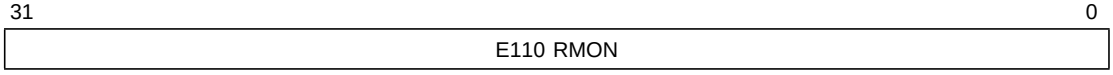
RMON_TX_DEFER
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3StatsDeferredTransmissions) [31:0]

Register: 0x340X.007C

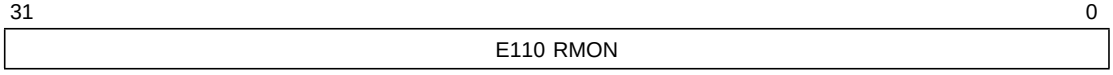
RMON_TX_LATE_COL
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3StatsLateCollisions) [31:0]

Register: 0x340X.0080

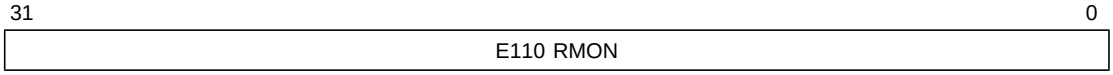
RMON_SINGLE_COL
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3StatsSingleCollisionFrames) [31:0]

Register: 0x340X.0084

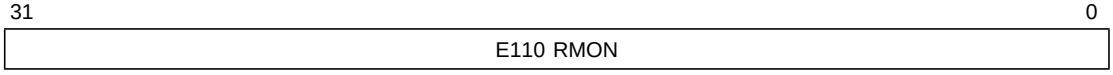
RMON_MULTI_COL
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3StatsMultiCollisionFrames) [31:0]

Register: 0x340X.0088

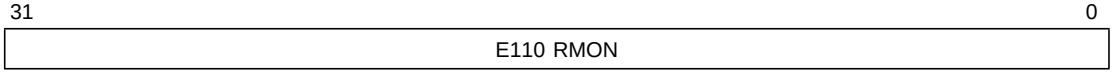
RMON_XSIVE_COL
CPU:R/W
Default:0x0000.0000



E110 RMON (dot3StatsExcessiveCollisions) [31:0]

Register: 0x340X.008C

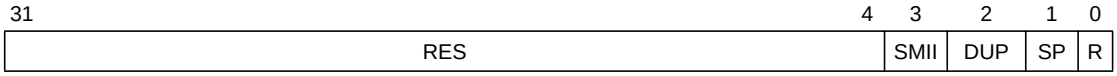
RMON_TOTAL_COLLISION
CPU:R/W
Default:0x0000.0000



E110 RMON (etherStatsCollisions) [31:0]

Register: 0x340X.0800

E110_SMII_STATUS
CPU:R
Default:0x0000.000x



RES **Reserved** **[31:4]**

SMII **SMII_Link** **3**

SMII	Description
0	Link is down
1	Link is up

DUP **Duplex** **2**

DUP	Description
0	Half duplex
1	Full duplex

SP **Speed** **1**

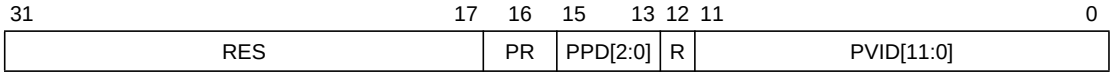
SP	Description
0	10 MHz
1	100 MHz

R **Reserved** **0**

This bit is reserved.

Register: 0x340X.0810

PVID_PPD (E110 Port Default VLAN and Priority)
CPU:R/W
Default:0x000.00xx



RES **Reserved** **[31:17]**
These bits are reserved.

PR **Priority Override** **16**
Priority override control. When PR is set, all packets received on the port will be given the priority in the PPD field of this register, even if they are tagged with a different priority.

PPD[2:0] **Port Priority** **[15:13]**
PPD[2:0] is a 3-bit priority tag that is assigned on a per port basis by the CPU. The CPU can use this priority tag to increase the priority of an originally untagged packet on a tagged link. It can also be used to replace the priority field of a tagged packet when the priority override bit (PR) is set for the input port.

R **Reserved** **12**
This bit is reserved.

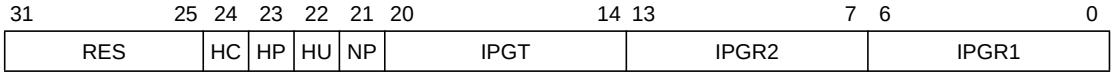
PVID[11:0] **Port VLAN ID** **[11:0]**
Each port has an associated default VLAN ID default of 1, or can be programmed to any one of 4096 VLAN IDs. The PVID field is transferred to bits 11:0 of the first four bytes of the packet buffer.

The 12-bit VLAN ID specifies membership in one of 4096 possible VLANs.

The default value of PVID[11:0] is PORT_ID.

Register: 0x340X.0820

E110_MAC_CFG_1
CPU:R/W
Default:0x0004.888B



RES **Reserved** **[31:25]**
These bits are reserved.

HST_CRCEN **Generate and Pad CRC** **24**
The MAC automatically generates the CRC on outgoing packets and checks CRC on incoming packets.

HST_PADEN **PAD Enable** **23**
The MAC automatically pads undersize packets with sufficient bytes of 0 such that the minimum packet size of 64 bytes is maintained during transmit packets.

HUGEN **Huge Enable** **22**
This bit must be cleared to 0, which disables packets larger than 1522 bytes from being processed.

NOPRE **No Preamble** **21**
Setting the NOPRE bit disables the MAC from generating a preamble.

Note: this may result in unreliable operation.

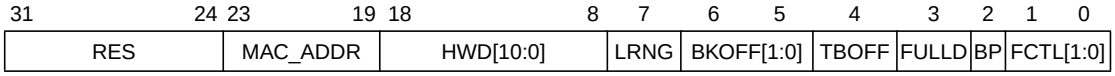
IPGT **Interpacket Gap Control** **[20:14]**
IPGT sets the interpacket gap value for back-to-back packets. The default is 0x12.

IPGR2 **Interpacket Gap Control** **[7:13]**
IPGR2 sets the interpacket gap value 2 for non back-to-back packets. The default is 0x11.

IPGR1 **Interpacket Gap Control** **[6:0]**
IPGR1 sets the interpacket gap value 1 for non back-to-back packets. The default is 0x0B.

Register: 0x340X.0830

E110_MAC_CFG_2
CPU:R/W
Default:0x00xx.xx00



RES **Reserved** **[31:24]**
These bits are reserved.

MAC_ADDR **MAC Address** **[23:19]**
The default value for this field is the PORT_ID.

HWD[10:0] **Host Write Data** **[18:8]**
This field is used to generate a random number sequence that is used in collision backoff timing. The default is 0b000000, PORT_ID.

LRNG **Load Random Number Generator** **7**
When this bit is 1, it indicates that the HWD[10:0] field is to be updated.

BKOFF[1:0] **Backoff Limit** **[6:5]**

BKOFF[1:0]	Description
0b00	Range = 0 to 1023
0b01	Range = 0 to 255
0b10	Range = 0 to 15
0b11	Range = 0 to 1

TBOFF **Transmit Backoff** **4**

TBKOFF	Description
0	Backoff after collision (default)
1	No backoff after collision

FULLD **Full Duplex Mode** **3**

FULLD	Description
0	Port operates in half duplex (default)
1	Port operates in full duplex

BP **Backpressure Select** **2**

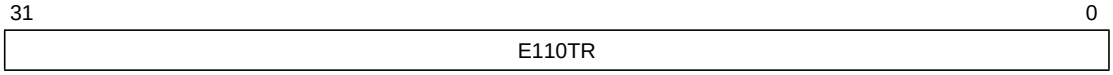
BP	Description
0	Back pressure is not used for flow control in half-duplex mode
1	Use back pressure to do flow control if port is in half-duplex mode

FCTL[1:0] **FLOW_CTL** **[1:0]**

FCTL[1:0]	Description
0b00	No flow control
0b01	Transmit only flow control
0b10	Receive only flow control
0b11	Both transmit and receive flow control

Register: 0x340X.1000

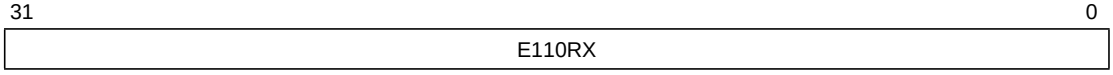
E110_TX_FIFO
CPU:R/W
Default:0x0000.0000



E110TR	**E110 Transmit FIFO (base)**	**[31:0]**

Register: 0x340X.1800

E110_RX_FIFO
CPU:R/W
Default:0x0000.0000



E110RX	**E110 Receive FIFO (base)**	**[31:0]**

8.8 1000 Mbits/s Port Registers

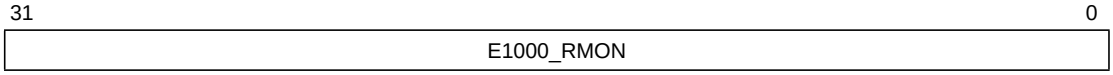
Ports 25 and 26 are Gigabit ports that each have a collection of registers detailed in the following pages. The register operation is the same for each port, and the base address of each register is listed in the table below.

Table 8.3 1000 Mbits/s Port Base Addresses

Port Number	Base Address
25	3418.0000 (only if 1000 Mbits/s operation is enabled)
26	3419.0000 (only if 1000 Mbits/s operation is enabled)

Register: 0x341X.0000

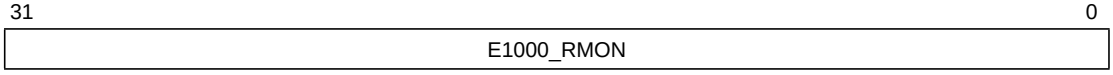
RMON_TOTAL_COUNT
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsOctets) [31:0]

Register: 0x341X.0004

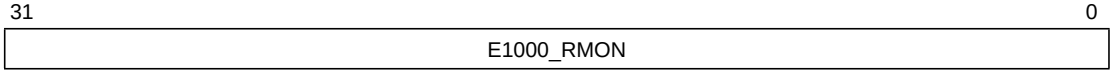
RMON_LENGTH64
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsPkts64Octets) [31:0]

Register: 0x341X.0008

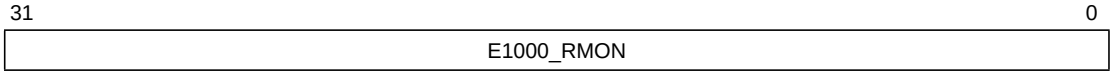
RMON_LENGTH65
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsPkts65to127Octets) [31:0]

Register: 0x341X.000C

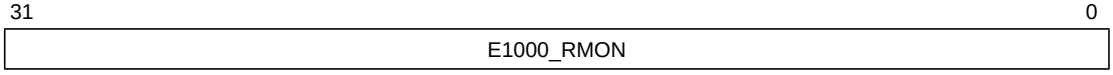
RMON_LENGTH128
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsPkts128to255Octets) [31:0]

Register: 0x341X.0010

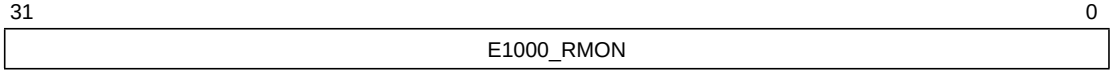
RMON_LENGTH256
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsPkts256to511Octets) [31:0]

Register: 0x341X.0014

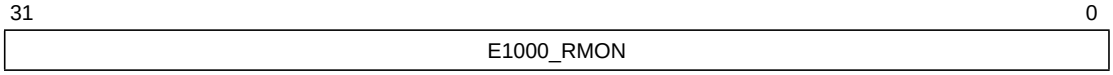
RMON_LENGTH512
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsPkts512to1023Octets) [31:0]

Register: 0x341X.0018

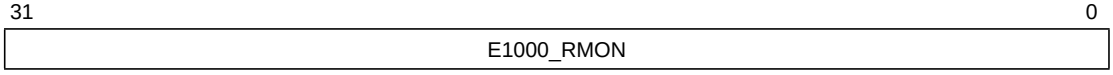
RMON_LENGTH1024
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsPkts1024to1518Octets) [31:0]

Register: 0x341X.001C

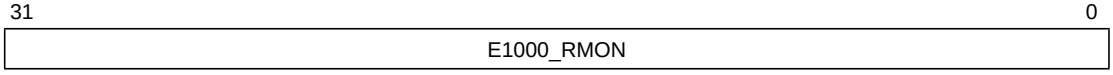
RMON_LENGTH1519
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsPkts1519to15220Octets) [31:0]

Register: 0x341X.0020

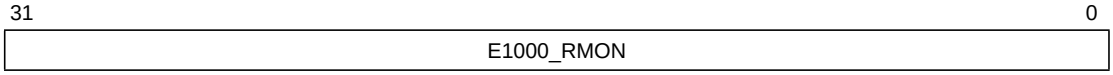
RMON_SYMBOL_ERROR
CPU:R/W
Default:0x0000.0000



E1000_RMON (dot3StatsSymbolErrors) [31:0]

Register: 0x341X.0024

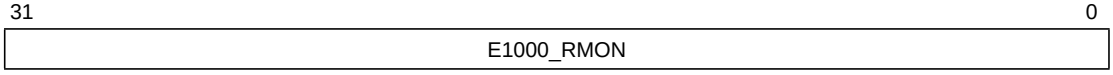
RMON_CRC_ERROR
CPU:R/W
Default:0x0000.0000



E1000_RMON (dot3StatsFCSErrors) [31:0]

Register: 0x341X.0028

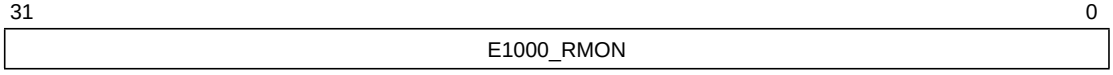
RMON_UNDERSIZE
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsUndersizePkts) [31:0]

Register: 0x341X.002C

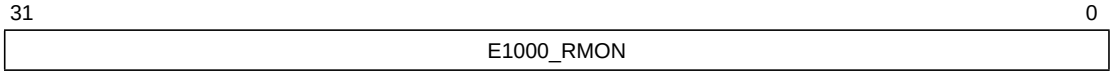
RMON_OVERSIZE
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsOversizePkts) [31:0]

Register: 0x341X.0030

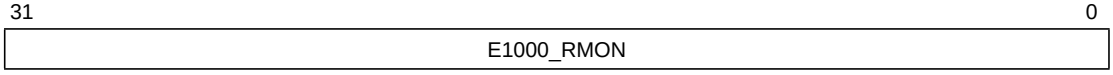
RMON_JABBER
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsJabbers) [31:0]

Register: 0x341X.0034

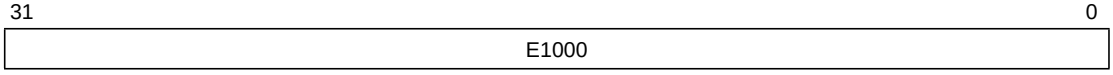
RMON_FRAGMENT
CPU:R/W
Default:0x0000.0000



E1000_RMON (etherStatsFragments) [31:0]

Register: 0x341X.0038

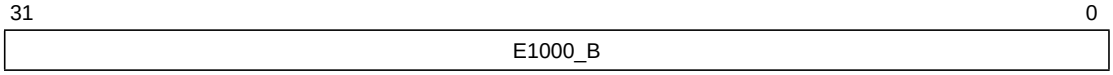
RMON_RX_COUNT
CPU:R/W
Default:0x0000.0000



E1000_RMON (ifInOctets) [31:0]

Register: 0x341X.003C

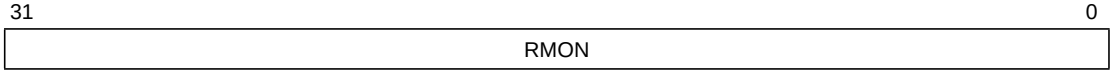
RMON_RX_BRDCAST
CPU:R/W
Default:0x0000.0000



E1000_RMON (ifInBroadicastPkts) [31:0]

Register: 0x341X.0040

RMON_RX_MLTCAST
CPU:R/W
Default:0x0000.0000



E1000_RMON (ifInMulticastPkts) [31:0]

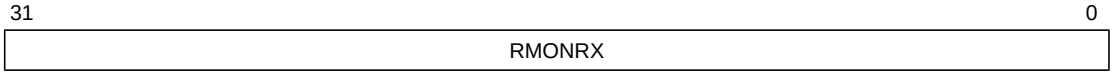
Register: 0x341X.0044

RMON_RX_UNICAST
CPU:R/W
Default:0x0000.0000

E1000_RMON (ifInUcastPkts) [31:0]

Register: 0x341X.0048

RMON_RX_PAUSE
CPU:R/W
Default:0x0000.0000



E1000_RMON (dot3PauseFrames) [31:0]

Register: 0x341X.004C

RMON_RX_UNKNOWN
CPU:R/W
Default:0x0000.0000

E1000_RMON (dot3ControlInUnknownOpcodes) [31:0]

Register: 0x341X.0050

RMON_RX_OVERFLOW
CPU:R/W
Default:0x0000.0000

E1000_RMON (dot3StatsInternalMacReceiveErrors) [31:0]

Register: 0x341X.0054

RMON_FALSE_CARRIER
CPU:R/W
Default:0x0000.0000

E1000_RMON (ifMauFalseCarriers) [31:0]

Register: 0x341X.0058

RMON_RX_PKT_DROP
CPU:R/W
Default:0x0000.0000

E1000_RMON (RXDroppedPackets) [31:0]

Register: 0x341X.005C

RMON_TX_COUNT
CPU:R/W
Default:0x0000.0000

E1000_RMON (ifOutOctets) [31:0]

Register: 0x341X.0060

RMON_TX_UNDER_RUN
CPU:R/W
Default:0x0000.0000

E1000_RMON (dot3StatsInternalMacTransmitError) [31:0]

Register: 0x341X.0064

RMON_TX_BRDCAST
CPU:R/W
Default:0x0000.0000

E1000_RMON (ifOutBroadcastPkts) [31:0]

Register: 0x341X.0068

RMON_TX_MLTCAST
CPU:R/W
Default:0x0000.0000

E1000_RMON (ifOutMulticastPkts) [31:0]

Register: 0x341X.006C

RMON_TX_UNICAST
CPU:R/W
Default:0x0000.0000

E1000_RMON (ifOutUnicastPkts) **[31:0]**

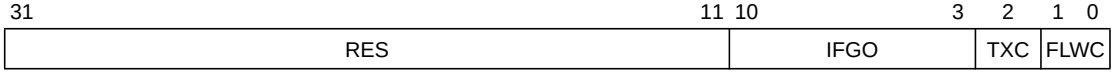
Register: 0x341X.0070

RMON_TX_PAUSE
CPU:R/W
Default:0x0000.0000

E1000_RMON (dot3OutPauseFrames) **[31:0]**

Register: 0x341X.0800

GIG_CFG
CPU:R/W
Default:0x0000.019C



RES **Reserved** **[31:11**
These bits are reserved.

IFGO[7:0] **Interframe Gap Offset** **[10:3]**
These bits control the receive to transmit interframe gap and the transmit to transmit interframe gap. The default value of 0b0011.0011 will give the IEEE 802.3 standard value of 0.096 µs (96 bit times).

TXC **Transmit Carrier Sense Defer** **2**

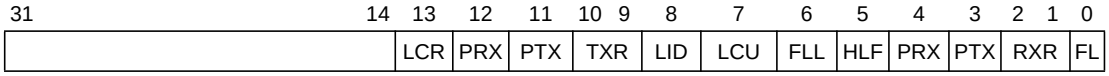
When set the MAC defers to its own carrier sense on transmit to produce the proper interpacket gap. When cleared, the MAC ignores its own carrier sense and transmits back to back packets with the smallest interpacket gap possible.

FLWC[1:0] **FLOW_CTL** **[1:0]**

FLWC[1:0]	Flow Control Action
0b00	No action
0b01	Assert 802.3X Pause Frame
0b10	React to 802.3X Pause Frame
0b11	Assert and react to 802.3 Pause Frames

Register: 0x341X.0804

GIG_PCS_HW_CFG (Gigabit PCS Hardware Link Configuration)
CPU:R/W
Default:0x0000.1840



RES **Reserved** **[31:14]**
Reserved. Software must write these bits as 0. The value is undefined on reads.

LCR **LC_RESTART** **13**
When written as 1, then 0, this bit restarts AutoNegotiation.

When read, this bit returns the last value written.

PRX **PAUSE_RX_CAPABLE** **12**
When written as 1, this bit pauses receive capability (PAUSE).

When read, this bit returns the last value written.

PTX **PAUSE_TX_CAPABLE** **11**
When written as 1, this bit pauses transmit capability (ASM_DIR).

When read, this bit returns the last value written.

TXR[1:0] **TX_REMOTE_FAULT** **[10:9]**
When written as 1, this bit causes a remote fault to be transmitted.

When read, this bit returns the last value written.

LID **LINK_INIT_DONE** **8**
Writes to this bit are ignored.

When read, this bit indicates if link initialization is complete.

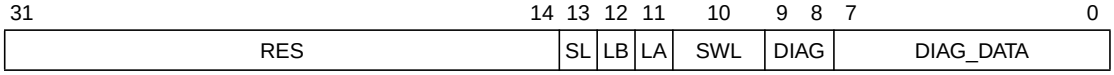
LCU **LC_UNRESOLVED** **7**
Writes to this bit are ignored.

When read, this bit indicates if the link initialization completed at least once, and the most recent attempt indicates the two stations are incompatible.

FLL	**FULL_DUPLEX_CHOSEN** Writes to this bit are ignored. When read, this bit indicates if full duplex mode has been negotiated.	**6**
HLF	**HALF_DUPLEX_CHOSEN** Writes to this bit are ignored. When read, this bit indicates if half-duplex mode has been negotiated.	**5**
PRX	**PARTNER_RX_PAUSE** Writes to this bit are ignored. When read, this bit indicates if the link partner has pause receive capability.	**4**
PTX	**PARTNER_TX_PAUSE** Writes to this bit are ignored. When read, this bit indicates if the link partner has pause transmit capability.	**3**
RXT[1:0]	**RX_REMOTE_FAULT** Writes to this bit are ignored. When read, this bit indicates a received remote fault indication.	**[2:1]**
FL	**FAST_LINK** When written, this bit initiates fast link timer mode (for testing purposes). When read, this bit returns the last value written.	**0**

Register: 0x341X.0808

GIG_PCS_SW_CFG (Gigabit PCS Software Link Configuration)
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:14]**
Software must write these bits as 0. The value is undefined on reads.

SL **SOFTWARE_LINK_INIT** **13**
When written as 1, then 0, this bit initiates software-based link initialization.

When read, this bit returns the last value written.

LB **LOOP_BACK_AND PCS_RESET** **12**
When written as 1, this bit initiates PCS loopback. When this bit transitions from 0 to 1, it automatically resets the PCS to get it into the loopback mode. Setting this bit when the bit is already 1 has no effect.

When read, this bit returns the last value written.

LA **LINK_AVAILABLE** **11**
When written as 1, this bit instructs the PCS to enter the transmit state and make the link available.

When read, this bit returns the last value written.

SWL **SW_LINKINIT_DONE** **10**
When set, this bit indicates that software link initialization is completed.

When read, this bit returns the last value written.

DIAG[1:0] **DIAG_MODE** **[9:8]**
This field is written to specify the transmit diagnostic mode.

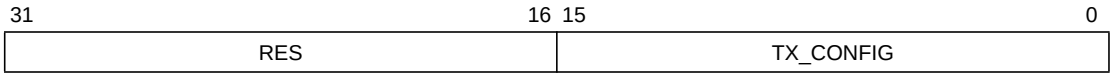
When read, this field returns the last value written.

DIAG[7:0] **DIAG_DATA** **[7:0]**
When written, this field loads the diagnostic data.

When read, this field returns the last value written.

Register: 0x341X.080C

GIG_PCS_SW_TXCFG (Gigabit PCS Transmit Configuration)
CPU:R/W
Default:0x0000.0000



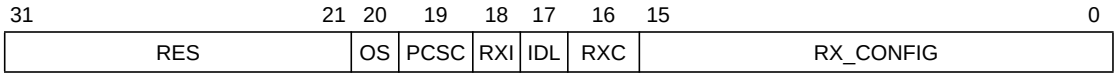
RES **Reserved** **[31:16]**
Software must write these bits as 0. The value is undefined on reads.

TX_CONFIG **Transmit Configuration** **[15:0]**
This field may be written with the transmit configuration value used in the IEEE 802.3z AutoNegotiation process.

When read, this field returns the last value written.

Register: 0x341X.0810

GIG_PCS_SW_STAT (Gigabit PCS Software Link Status)
CPU:R
Default:0x0000.0000



RES **Reserved** **[31:21]**
Software must write these bits as 0. The value is undefined on reads.

OS **OUT_OF_SYNC** **20**
Writes to this bit are ignored.

When read as 1, this bit indicates an excessive number of errors have been detected in the 10-bit symbol stream.

PCSC **PCS_CONFIG_CHANGE** **19**
Writes to this bit are ignored.

When read as 1, this bit indicates that the value in the RX_CONFIG_REG has changed.

RXI **RX_INVALID** **18**
Writes to this bit are ignored.

When read as 1, this bit indicates that the PCS has received an invalid code group.

IDL **IDLE_MATCH** **17**
Writes to this bit are ignored.

When read as 1, this bit indicates that three consecutive Idle Ordered Sets have been received.

RXC **RX_CONFIG_VALID** **16**
Writes to this bit are ignored.

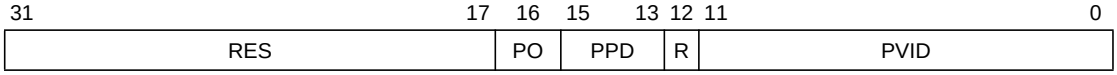
When read as 1, this bit indicates that the contents of the RX_CONFIG_REG are valid.

RX_CONFIG[15:0]
RX_CONFIG_REG **[15:0]**
Writes to this bit are ignored.

When read as 1, this bit indicates that the PCS received the last configuration code word.

Register: 0x341X.0814

PVID_PPD (Gigabit Port Default VLAN and Priority)
CPU:R/W
Default:0x0000.0001



RES **Reserved** **[31:17]**
These bits are reserved.

PR **Priority Override** **16**
Priority override control. When PR is set, all packets received on the port will be given the priority in the PPD field of this register, even if they are tagged with a different priority.

PPD[2:0] **Port Priority** **[15:13]**
PPD[2:0] is a three-bit priority tag that is assigned on a per port basis by the CPU. The CPU can use this priority tag to increase the priority of an originally untagged packet on a tagged link. It can also be used to replace the priority field of a tagged packet when the priority override bit (PR) is set for the input port.

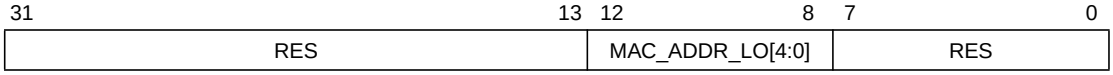
R **Reserved** **12**
These bits are reserved.

PVID[11:0] **Port VLAN ID** **[11:0]**
Each port has an associated default VLAN ID default of 1, or can be programmed to any one of 4096 VLAN IDs. The PVID field is transferred to bits 11:0 of the first four bytes of the packet buffer.

The 12-bit VLAN ID specifies membership in one of 4096 possible VLANs.

Register: 0x341X.0818

MAC_ADDR_LO (Gigabit Port MAC Address Bits[4:0])
CPU:R/W
Default:Port_ID



RES **Reserved** **[31:13**
These bits are reserved.

MAC_ADDR_LO[4:0] **[12:8]**
The default value is the Port_ID

RES **Reserved** **[7:0]**
These bits are reserved.

Register: 0x341X.1000

GIGA_TX_FIFO (Gigabit Transmit FIFO (base))
CPU:R/W
Default:00x0000.0000

TXFIFO **Transmit FIFO** **[31:0]**
The transmit FIFO is 192 bytes deep (48 32-bit words). See the FIFO chapter for operational details.

Register: 0x341X.1800

GIGA_RX_FIFO (Gigabit Receive FIFO (base))
CPU:R/W
Default:0x0000.0000

RXFIFO **Receive FIFO** **[31:0]**
The receive FIFO is 1008 bytes deep. See the FIFO chapter for operational details.

Register: 0x341X.2000–0x341X.2230

GIGA_EXTRACTOR_REGS (Gigabit Extractor Registers)
CPU:R
Default:0x0000.0000

31	0
EXT_PKTX[31:0]	
EXT_PKTX[63:32]	
EXT_PKTX[95:64]	
EXT_PKTX[127:96]	
EXT_PKTX[141:128]	

The extracted packet information is 18 packets x 142 bits.

Each Gigabit port can store extracted information for up to 18 packets, with 142 bits per packet. The following table shows the packet names and addresses of each extracted packet.

Table 8.4 Extracted Gigabit Port Packets

Extracted Packet Name	Bit Field	Base Address (bits 9:5)
EXT_PKT_1	[31:0]	2000
	[63:32]	2004
	[95:64]	2008
	[127:96]	200C
	[141:128]	2010
EXT_PKT_2	[31:0]	2200
	[63:32]	2204
	[95:64]	2208
	[127:96]	220C
	[141:128]	2210
EXT_PKT_3	[31:0]	2220
	[63:32]	2224
	[95:64]	2228
	[127:96]	222C
	[141:128]	2230

(Sheet 1 of 4)

Table 8.4 Extracted Gigabit Port Packets (Cont.)

Extracted Packet Name	Bit Field	Base Address (bits 9:5)
EXT_PKT_4	[31:0]	2020
	[63:32]	2024
	[95:64]	2028
	[127:96]	202C
	[141:128]	2030
EXT_PKT_5	[31:0]	2060
	[63:32]	2064
	[95:64]	2068
	[127:96]	206C
	[141:128]	2070
EXT_PKT_6	[31:0]	20E0
	[63:32]	20E4
	[95:64]	20E8
	[127:96]	20EC
	[141:128]	20F0
EXT_PKT_7	[31:0]	21E0
	[63:32]	21E4
	[95:64]	21E8
	[127:96]	21EC
	[141:128]	21F0
EXT_PKT_8	[31:0]	21C0
	[63:32]	21C4
	[95:64]	21C8
	[127:96]	21CC
	[141:128]	21D0
EXT_PKT_9	[31:0]	21A0
	[63:32]	21A4
	[95:64]	21A8
	[127:96]	21AC
	[141:128]	21B0

(Sheet 2 of 4)

Table 8.4 Extracted Gigabit Port Packets (Cont.)

Extracted Packet Name	Bit Field	Base Address (bits 9:5)
EXT_PKT_10	[31:0]	2160
	[63:32]	2164
	[95:64]	2168
	[127:96]	216C
	[141:128]	2170
EXT_PKT_11	[31:0]	2140
	[63:32]	2144
	[95:64]	2148
	[127:96]	214C
	[141:128]	2150
EXT_PKT_12	[31:0]	2180
	[63:32]	2184
	[95:64]	2188
	[127:96]	218C
	[141:128]	2190
EXT_PKT_13	[31:0]	2100
	[63:32]	2104
	[95:64]	2108
	[127:96]	210C
	[141:128]	2110
EXT_PKT_14	[31:0]	2120
	[63:32]	2124
	[95:64]	2128
	[127:96]	212C
	[141:128]	2130
EXT_PKT_15	[31:0]	20A0
	[63:32]	20A4
	[95:64]	20A8
	[127:96]	20AC
	[141:128]	20B0

(Sheet 3 of 4)

Table 8.4 Extracted Gigabit Port Packets (Cont.)

Extracted Packet Name	Bit Field	Base Address (bits 9:5)
EXT_PKT_16	[31:0]	2080
	[63:32]	2084
	[95:64]	2088
	[127:96]	208C
	[141:128]	2090
EXT_PKT_17	[31:0]	20C0
	[63:32]	20C4
	[95:64]	20C8
	[127:96]	20CC
	[141:128]	20D0
EXT_PKT_18	[31:0]	2040
	[63:32]	2044
	[95:64]	2048
	[127:96]	204C
	[141:128]	2050

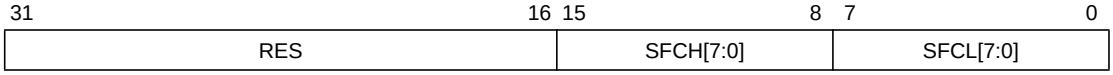
(Sheet 4 of 4)

8.9 ABIT Registers

The base address location of the ABIT registers is 0x3426.0000.

Register: 0x3426.0000

System Flow Control
CPU:R/W
Default:0x0000.0000



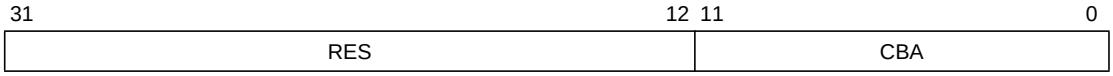
RES **Reserved** **[31:16]**
These bits are reserved.

SFCH [7:0] **System Flow Control High Threshold** **[15:8]**
This 8-bit field is multiplied by 4 to form the 10-bit global register. This register is used to terminate flow control type 1 when CBA is equal to or greater than this register. The default value is 0x00.

SFCL[7:0] **System Flow Control Low Threshold** **[7:0]**
This 8-bit field is multiplied by 4 to form the 10-bit global register. Flow Control Type 1 is initiated when CBA becomes equal to or less than this register. The default value is 0x00.

Register: 0x3426.0004

CBA
CPU:R/W
Default:0x0000.0FFF

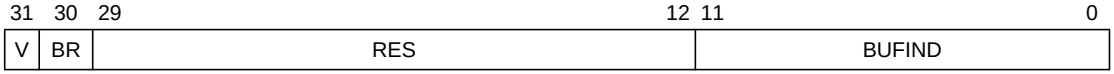


RES **Reserved** **[31:12]**
These bits are reserved.

CBA[11:0] **Count of Buffers Available** **[11:0]**
This 12-bit field points to the next available buffer in the memory. CBA is decremented every time the ABIT index is allocated, and incremented every time the index has been returned to ABIT. Normally, the CPU should not write this register at any time.

Register: 0x3426.0008

CPU_BUF_REQ
CPU:R/W
Default:0x000.0000



The CPU uses this register to request a buffer from the ABIT.

V **Valid** **31**
This bit is set to 1 when a new buffer has been allocated to the CPU by updating bits [11:0] with an index pointer. This bit is cleared when the CPU reads this register.

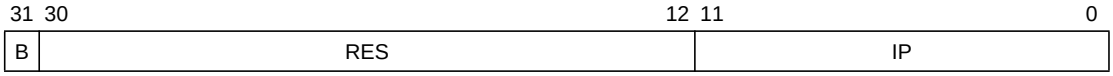
BR **Buf_REQ** **30**
The CPU sets this bit when it requests a new buffer.

RES **Reserved** **[29:12]**
These bits are reserved.

BUFIND[11:0] **Buffer Index** **[11:0]**
This 12-bit value indicates which buffer the CPU should use for the current request.

Register: 0x3426.000C

BUF_RET (Buffer Return)
CPU:R/W
Default:0x00.0000



This register is used by the CPU to return a buffer back to ABIT. When the CPU wants to return a buffer it writes to the index_pointer at bits [11:0].

B **Busy** **31**
Software sets the B bit to 1 when a new index pointer is loaded for returning to the ABIT. Hardware clears the bit when the index pointer is updated in the ABIT. Software needs to poll this bit to ensure it is 0 before writing to this register.

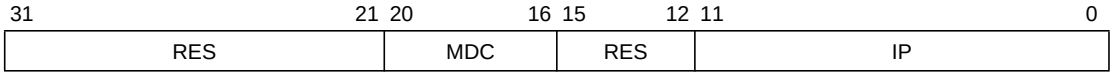
RES **Reserved** **[30:12]**
These bits are reserved.

IP[11:0] **Index Pointer** **[11:0]**
The IP field contains the index of the buffer being returned.

Register: 0x3426.4000–0x3426.7FFF

ABIT Memory Location
CPU:R/W
Default:0x0000.0000

The ABIT appears as a 32-bit wide memory structure containing 4096 entries. The first location, location 0, is positioned as shown in the register table below. Each 32-bit entry has the format shown. This structure only supports full 32-bit accesses from the CPU. Byte accesses are not supported. These locations are read or written as 32-bit words. Bits [31:21] and bits [15:12] are ignored during writes and return as zeros during reads. The ABIT is organized as 17-bit words. Each entry has a 12-bit index pointer (IP) and 5-bit multidestination count (MDC).



RES **Reserved** **[31:21]**
These bits are reserved.

MDC[4:0] **Multidestination Count** **[20:16]**
The MDC field contains the number of destinations that will transmit the associated packet. The CPU should initialize MDC to 0b00001 at all locations.

RES **Reserved** **[15:12]**
These bits are reserved.

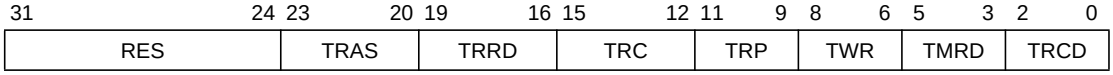
IP[11:0] **Index Pointers** **[11:0]**
The CPU should initialize all IP fields so that adjacent indexes are not positioned in the same bank initially. A sequential count of 0, 2, 4,.....4094, 1, 3, 5, 7....4095 satisfies this requirement.

8.10 Memory Controller Registers

The Memory Controller registers are located at base address 0x3427.0000.

Register: 0x3427.0000

SDRAM_TIMING_REG (SDRAM Timing)
CPU:R/W
Default:0x0006.389B



RES **Reserved** **[31:24]**
These bits are reserved.

TRAS[3:0] **TRAS RAS to Precharge Delay** **[23:20]**
Default = 0b0110

TRRD[3:0] **RAS to RAS Delay** **[19:16]**
Default = 0b0011

TRC[3:0] **Time from Refresh to Next Active Cycle** **[15:12]**
Default = 0b1000

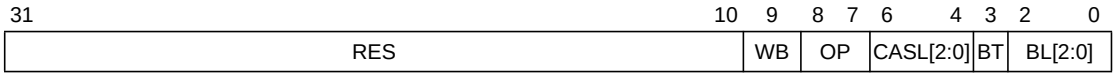
TRP[2:0] **Time Precharge to Active Cycle** **[11:9]**
Default = 0b011

TWR[2:0] **Write Recovery Time** **[8:6]**
Default = 0b010

TMRD[2:0] **Mode Reg Set to Active** **[5:3]**
Default = 0b011

TRCD[2:0] **Active to Read/Write Delay** **[2:0]**
Default = 0b011

Register: 0x3427.0004
SDRAM_MODE_REG (SDRAM Mode)
CPU:R/W
Default:0x0000.0033



RES **Reserved** **[31:10]**

WB **Write Burst** **9**

WB Bit	Definition
0	Programmed Burst (Default, fixed at 0)
1	Single Write (not supported)

OP[1:0] **OP Mode** **[8:7]**
The only value supported is 0b00, which is standard operation.

CASL[2:0] **CAS Latency** **[6:4]**

CASL[2:0]	Definition
0b000	Reserved
0b001	1 Clock Latency
0b010	2 Clocks Latency
0b011	3 Clocks Latency
0b101–0b111	Reserved

BT **Burst Type** **3**

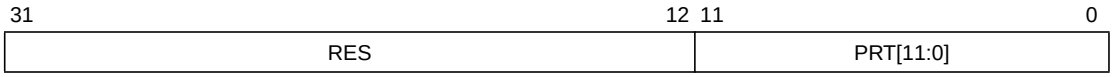
BT Bit	Definition
0	Sequential (Default, fixed at 0)
1	Interleave (not supported)

BL[2:0] **Burst Length** **[2:0]**

BL[2:0]	Definition
0b000	1 Word Burst
0b001	2 Words Burst
0b010	4 Words Burst
0b011	8 Words Burst
0b101–0b110	Full Page
0b111	Reserved

Register: 0x3427.0008

SDRAM_REFRESH_REG (SDRAM Refresh)
CPU:R/W
Default:0x0000.07D1

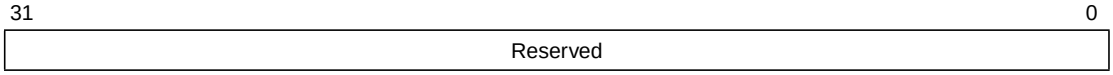


RES **Reserved** **[31:12]**
These bits are reserved.

PRT[11:0] **Programmable Refresh Timer** **[11:0]**
The value of 0x7D1 corresponds to 16 µs, which is the required SDRAM refresh period. To program another value divide the time required by 8 ns (period of the 125 MHz clock).

Register: 0x3427.000C

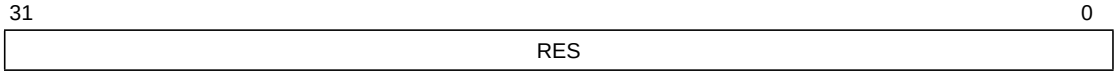
Reserved
CPU:R/W
Default:0x0000.0000



This register is reserved. Do not use it.

Register: 0x3427.0010

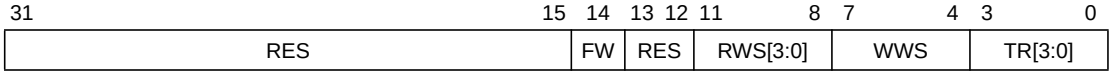
RESERVED
CPU:R/W
Default:0x0000.0000



This register is reserved. Do not use it.

Register: 0x3427.0020

FLASH_CONFIG (Flash Configuration)
CPU:R/W
Default:Board Configuration Pins



RES **Reserved** **[31:15]**
These bits are reserved.

FW **FLASH Write Enable** **14**
The FLASH can be programmed only when the FW bit is 1 (default). When the bit is 0, the FLASH is not writable and any attempt to write to the FLASH is ignored. This is to protect any unwanted writes to FLASH.

RES **Reserved** **[13:12]**
These bits are reserved.

RWS[3:0] **Read Wait Cycles** **[11:8]**
These four bits are sampled from the SDRAM[93:90] data pins at power-up or reset.

WWS[3:0] **Write Wait Cycles** **[7:4]**
These four bits are sampled from the SDRAM[93:90] data lines at power-up or reset.

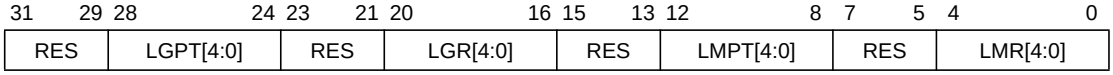
TR[3:0] **Transaction Turnaround Clocks** **[3:0]**
The TR[3:0] field is latched from the SDRAM[71:68] data during reset.

8.11 DMA Controller Registers

The DMA Controller registers are located at base address 0x3428.0000.

Register: 0x3428.0000

LAST_TX_STATUS (Last Transmit Status)
CPU:R
Default:0x3818.2000



RES **Reserved** **[31:29]**
These bits are reserved.

LGPT[4:0] **Last Gigabit Port Transmit** **[28:24]**
This field indicates the Last Gigabit port that was granted a read access to memory.

RES **Reserved** **[23:21]**
These bits are reserved.

LGR[4:0] **Last Gigabit Receive** **[20:16]**
This field indicates the Last Gigabit Port to be granted a write access to memory.

RES **Reserved** **[15:13]**
These bits are reserved.

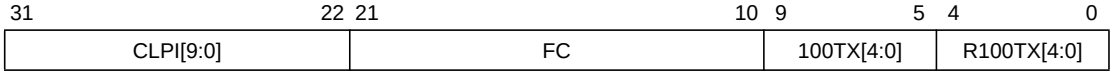
LMPT[4:0] **Last Megabit Port Transmit** **[12:8]**
This field indicates the Last 10/100 port that was granted a read access to memory.

RES **Reserved** **[7:5]**
These bits are reserved.

LMR[4:0] **Last Mega Receive** **[4:0]**
This field indicates the Last 10/100 port to be granted a write transaction to memory.

Register: 0x3428.0080

ARBITRATOR_FAIRNESS (DMA Arbitrator Fairness)
CPU:R/W
Default:0x9A00.0318



CLPI[9:0] **CPU Low Priority Interval** **[31:22]**
This 10-bit value is multiplied by 16 to form the number of 8 ns clock cycles that must expire, after the CPU is serviced, before the CPU priority is increased. Setting CPLI[9:0] to 0 disables this function. For programming a time value, divide the time value by 16 x 8 (128) and load the equivalent hexadecimal value in this field. The default value is 0x268.

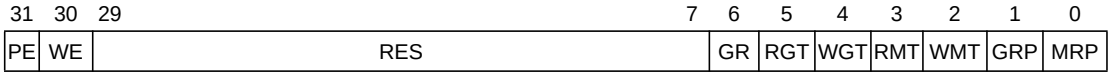
FC[11:0] **Fairness Count** **[21:10]**
This field is multiplied by 8.192 µs to create the time interval that the reduced 100TX count value will be valid. When the count expires, the system returns to a default value. The default value is 0x000.

100TX[4:0] **Default 100TX Count** **[9:5]**
This field determines the default number of concurrent 100TX transmits ever allowed to be active.

R100TX[4:0] **Reduced 100TX Count** **[4:0]**
This field determines the number of concurrent 100TX transmits allowed to be active after detecting a FIFO overrun on any port. When the current active count is greater than this value, no new transmits are granted. The value can range from 0 to 24.

Register: 0x3428.0088

PREFETCH_BUFF_EN (DMA Prefetch Buffer/Write Buffer Enable)
CPU:R/W
Default:0xC000.0000



PE **Prefetch Enable** **31**

PE Bit	Definition
0	Disable prefetch buffer
1	Enable prefetch buffer (default)

WE **Write Enable** **30**

WE Bit	Definition
0	Disable write buffer
1	Enable write buffer (default)

RES **Reserved** **[29:7]**
These bits are reserved.

GR **DMA High Priority for Gig Rx without Tx Port Check (1)** **6**

RGT **DMA High Priority for CPU Read with Gig Tx (1)** **5**

WGT **DMA High Priority for CPU Write with Gig Tx (1)** **4**

RMT **DMA High Priority for CPU Read with Meg Tx (1)** **3**

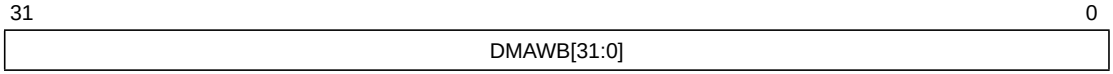
WMT **DMA High Priority for CPU Write with Meg Tx (1)** **2**

GRP **DMA High Priority for Gig Rx Port (1)** **1**

MRP **DMA High Priority for Mega Rx Port (1)** **0**

Register: 0x3428.0340–0x3428.039F

WRITE_BUFF (DMA Write Buffer)
CPU:R/W
Default:Unknown

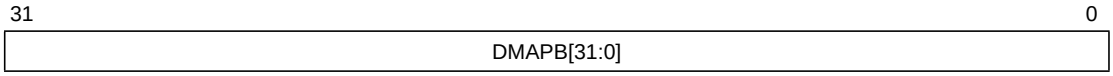


DMAWB[31:0] Write buffer for DMA [31:0]

The DMAWB[31:0] register set starts at 0x340 and ends at 0x39F. It is a 96-byte DMA memory.

Register: 0x3428.0640–0x3428.069F

PREFETCH_BUFF (DMA Prefetch Buffer)
CPU:R/W
Default:Unknown



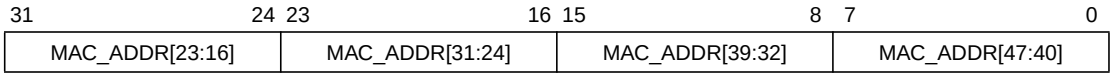
DMAPB **DMA Prefetch Buffer** **[31:0]**
The DMAPB[31:0] register set starts at 0x640 and ends at 0x69F. It is a 96-byte DMA memory.

8.12 ARL Registers and Memory Mapping

The ARL registers and memory mapping registers are located at base address 0x3425.0000.

Register: 0x3425.0000

ADDRESS (MAC Address)
CPU:R/W
Default:0xxxxx.xxxx

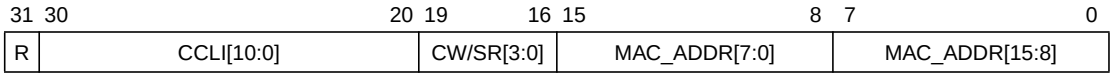


MAC_ADDR **MAC Address** **[31:0]**

This register contains the upper 32 bits of the MAC address in little endian format, as shown in the register table above.

Register: 0x3425.0004

ADDR_CMD (Commands and MAC Address)
CPU:R/W
Default:0x0xx0.xxxx



R **Reserved** **31**
This bit is reserved.

CCLI[10:0] **Collision Count/Location Index** **[30:20]**
This field contains the collision count. When the CPU issues a command of Return nth Hash Collision (CW[3:0] = 0b0101), it issues a write to location CCLI[2:0], to store the nth Hash Collision count. The Collision count result of a command is always returned in the HCN[2:0] field of register 0x3425.0008 (UC1/MC1).

When the CPU issues a command of Return Entry Given Location, (CW[3:0] = 0b0100), it issues a write to location CCLI[10:0], to store the Location Index. The Location Index result of a command is also returned from these same bits when the CPU reads them.

CW[3:0] **Command (*Write*)** **[19:16]**
This field contains the command issued from the CPU

CW[3:0]	Definition
0b0000	No operation
0b0001	Add
0b0010	Modify
0b0011	Return Entry Given Address
0b0100	Return Entry Given Location
0b0101	Return n^{th} Hash Collision
0b0110	Delete Entry
0b0111–0b1111	Reserved

SR[3:0] **Status (*Read*)** **[19:16]**

This field contains the command execution status.

SR[3:0]	Definition
0b0000	Completed successfully
0b0001	Command in progress
0b0010	Address not found
0b0011	Address could not be added, due to lack of free entry
0b0100	Address could not be added, because it exists already
0b0101	Invalid hash number.
0b0110–0b1110	Reserved for future use.
0b1111	Invalid command.

MAC_ADDR[15:0]

MAC Address **[15:0]**

This field contains the lower 16 bits of the MAC Address in little endian format as shown in the register table.

Register: 0x3425.0008

UC1/MC1 (ARL Unicast/Multicast Word 1)
CPU:R/W[12:0], R[31:11]
Default:0x0xx0.0xxx

31	30 … 20	19	18 … 16	15	14 … 13	12	11 … 7	6 … 5	4	3	2	1	0
R	LI	R	HCN	A	RES	U/PO	SRC/MC[4:0]	PCPU[1:0]	CE	PM	E/U	DA/U	A/U

This register contains the Unicast/Multicast Fields for the CPU command Interface.

R — **Unicast: Reserved** — **31**
Multicast: Reserved
This bit must be written as 0 and ignored on read.

LI[10:0] — **Unicast: Location Index (Read Only)** — **[30:20]**
Multicast: Location Index (Read Only)
The location Index result of a command in the range of 0–2047 is returned in this field.

R — **Reserved** — **19**
This bit must be written as 0 and ignored on read.

HCN[2:0] — **Unicast: Hash Collision Number (Read Only)** — **[18:16]**
Multicast: Hash Collision Number (Read Only)
The hash collision number (0–5) result of a command is returned in this field.

A — **Unicast: Addr[40] Group Bit (Read Only)** — **15**
Multicast: Addr[40] Group Bit (Read Only)
The A bit is bit 40 of the MAC address. It is the same as bit 0 of the ADDRESS register (0x3425.0000).

RES — **Unicast: Reserved** — **[14:13]**
Multicast: Reserved
These bits must be written as 0 and ignored on read.

U/P — **Unicast: Undefined** — **12**
Multicast: Priority Output (Boost)

SRC/MC[4:0] — **Unicast: SRC PORT[4:0]** — **[11:7]**
Multicast: Multicast Type
This field and the following fields of this register are written for Add and Modify commands, with respective

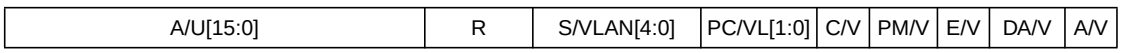
data depending on Unicast or Multicast. When a command completes, these fields contain the respective data, depending on Unicast or Multicast from the ARL Table Entry.

PCPU[1:0]	**Unicast: Priority CPU** **Multicast: Priority CPU**	**[6:5]**
CE	**Unicast: CPU** **Multicast: Enable Multicast Processing (EMP)**	**4**
PM	**Unicast: Port Mirror** **Multicast: Port Mirror**	**3**
E/U	**Unicast: Empty/Unused** **Multicast: Unused**	**2**
DA/U	**Unicast: Don't Age** **Multicast: Unused**	**1**
A/U	**Unicast: Age** **Multicast: Unused**	**0**

Register: 0x3425.000C

UC 2/MC 2 (ARL Unicast/Multicast Word 2)
CPU:R/W
Default:0xxxxx.0xxx

This register contains the Unicast/Multicast fields for the CPU command interface.



A/U[15:0] — **Unicast: Address {[7:0], [15:8]} (2nd Entry)** **[31:16]**
Multicast: Concatenation of {Unused MC Word 2 bit[31:19], Unused MC Word-2 bit[18:16]}

R — **Unicast: Reserved** **[15:12]**
Multicast: Reserved
Must be zero.

S/VLAN[4:0] — **Unicast: Source Port (2nd Entry)** **[11:7]**
Multicast: VLAN[11:7]

PC/VL[1:0] — **Unicast: Prior CPU (2nd Entry)** **[6:5]**
Multicast: VLAN[6:5]

C/V — **Unicast: CPU (2nd Entry)** **4**
Multicast: VLAN[4]

PM — **Unicast: Port Mirror (2nd Entry)** **3**
Multicast: VLAN[3]

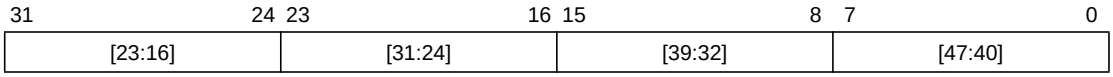
E/V — **Unicast: Empty (2nd Entry** **2**
Multicast: VLAN[2]

DA/V — **Unicast: Don't Age (2nd Entry)** **1**
Multicast: VLAN[1]

A/V — **Unicast: Age (2nd Entry)** **0**
Multicast: VLAN[0]

Register: 0x3425.0010

UC 3/MC 3 (ARL Unicast/Multicast Word 3)
CPU:R/W
Default:0xxxxx.xxxx



Contains the Unicast/Multicast Fields for the CPU command interface.

Unicast

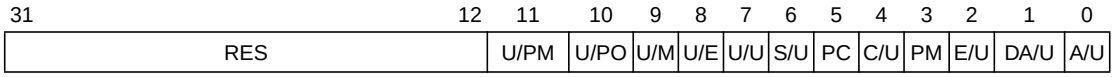
Address {[23:16], [31:24], [39:32], [47:40]}	**[31:0]**

Multicast

Unused Multicast Word 2 bits [20:16]	**[31:27]**
Port Mask [26:1]	**[26:1]**
Reserved	**0**

Register: 0x3425.0014

FIELD_SEL (Field Selector)
CPU:R/W
Default:0x0000.0000



Contains the Field Selector Bits for the CPU Command Interface.

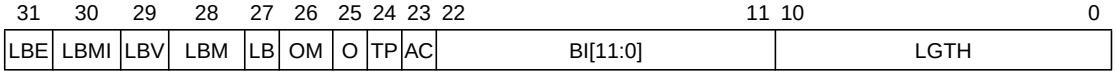
RES	**Unicast: Reserved** **Multicast: Reserved**	**[31:12]**
U/PM	**Unicast: Unused** **Multicast: Port Mask**	**11**
U/PO	**Unicast: Unused** **Multicast: Priority Output**	**10**
U/M	**Unicast: Unused** **Multicast: Multicast Type**	**9**
U/E	**Unicast: Unused** **Multicast: EMP**	**8**
UU	**Unicast: Unused** **Multicast: Unused Multicast word 2 bits {[59:47], [20:16], [15:13], [2:0]}**	**7**
S/U	**Unicast: Source Port** **Multicast: Unused**	**6**
PC	**Unicast: Priority_CPU** **Multicast: Priority_CPU**	**5**
C/U	**Unicast: CPU** **Multicast: Unused**	**4**
PM	**Unicast: Port Mirror** **Multicast: Port Mirror**	**3**
EU	**Unicast: Empty** **Multicast: Unused**	**2**

DA/U	**Unicast: Don't Age** **Multicast: Unused**	**1**
A/U	**Unicast: Age** **Multicast: Unused**	**0**

Register: 0x3425.0018

PKT_TX_1 (Transmit Descriptor First Word)
CPU:R/W
Default:0xx0xx.xxxx

This register contains the Transmit Descriptor First Word for CPU packet transmission.



LBE **LB_ERMON** **31**
When this bit is 1, looping back to the CPU due to Extended RMON is permitted.

LBMI **LB_MIRROR** **30**
When this bit is 1, looping back to the CPU due to Mirroring is permitted.

LBV **LB_VLAN** **29**
When this bit is 1, looping back to the CPU due to VLAN Table CPU bits is permitted.

LBM **LB_MCAST** **28**
When this bit is 1, looping back to the CPU due to this address being multicast and found in the table with type code nonzero or this multicast address not found in the table and UM_EN = 1 is permitted.

LB **LB_CPU_BIT** **27**
When this bit is 1, looping back to the CPU due to the CPU bit set in the address table for either Destination Address or Source Address is permitted. The default is 0.

OM **Outmask** **26**
The CPU sets this bit. The default is 0.

O **Own** **25**
When this bit is set, the ARL processes the request and clears this bit when it is done.

TP **Tagged Packet** **24**

This bit, when 1, indicates the CPU has placed a VLAN tag in the packet being transmitted. This bit performs the same function as the TP bit of each port. The default is 0.

AC **Add CRC** **23**

AC	Description
0	The software has already included the CRC that it wishes to be sent with the packet.
1	The packet needs to have CRC regenerated by the ASIC.

If the ASIC tags the packet on output (which can happen when TP = 0, OM = 0, and the packet goes out a tagged port), the CRC is regenerated regardless of this bit.

BI[11:0] **Buffer Index** **[22:11]**

BI[11:0] is the buffer index of the packet. The address of the buffer can be calculated by multiplying the buffer index by 1536, then adding the based address of the packet buffer space. This must be a buffer pointer obtained from the ABIT. The ASIC never returns this buffer to the ABIT. Instead, the CPU should check the ports for the completion of transmission of this packet and should return the buffer to the ABIT, if so desired.

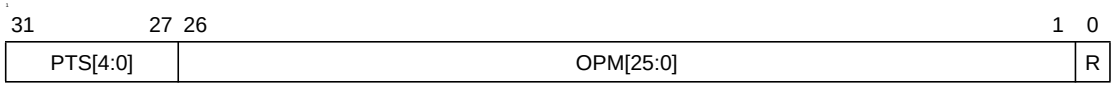
LGTH[10:0] **Length** **[10:0]**

LGTH[10:0] is the length of the packet, in bytes. It always includes the four bytes of CRC, regardless of the state of the AC bit. Bytes are counted starting with the first byte of the destination address. The length does not include the bytes at the start of the buffer before the packet.

Register: 0x3425.001C

PKT_TX_2 (Transmit Descriptor Second Word)
CPU:R/W
Default:0x00xx.xxx0

This register contains the Transmit Descriptor Second Word for CPU packet transmission.



PTS[4:0] **Packet Transmit Status** **[31:27]**
When the ARL clears bit 27 of PKT_TX_1, it indicates that the packet was successfully transmitted. This is the default value. When bit 27 is 1, it indicates that the packet was rejected.

PTS[3:0]	Definition
0b0000	Packet successfully transmitted
0b0001	Packet rejected
0b0010–0b1111	Reserved

OPM[25:0] **Output Port Mask** **[26:1]**
The OPM[25:0] field is effective when the OM bit of the PKT_TX_1 is set.

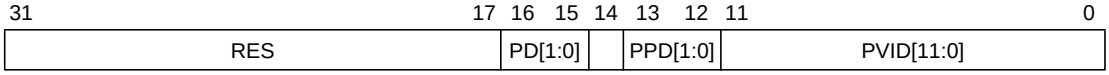
The OM bit enables the CPU to override the normal ASIC processing of VLAN, priority, and output port mask. When OM is set, no ports add or modify VLAN tags, and the output port mask (except for mirroring and Extended RMON) is specified with the OPM[25:0] field. When the OM bit is 0, the packet is transmitted using the specified PVID (VLAN_ID), and Priority (based on PPD and multicast Priority (Boost) from the ARL). Additionally, each port adds or removes tags as needed, depending on the ports configuration in the VLAN table.

R **Reserved** **0**
This bit must be zero.

Register: 0x3425.0020

PKT_TX 3 (Transmit Descriptor Third Word)
CPU:R/W
Default:0x0000.xxxx

This register contains the Transmit Descriptor Third Word for CPU packet transmission.



RES **Reserved** **[31:17]**
Must be zero.

PD[1:0] **Protocol Decode** **[16:15]**

PD[1:0]	Definition
0b00	Data packet (normal forwarding)
0b01	IGMP (send to CPU)
0b10	OSPF (send to CPU, flood)
0b11	PIM (send to CPU, flood)

When packets are received from the L64324 MACs, the port logic examines the data to detect if the packet is a multicast IGMP, PIM, OSPF, or normal data packet and encodes this information as shown in the table. For the CPU transmitted packets to be handled as per the protocol, software should provide this field in the correct state.

PPD[1:0] **Priority Default** **[13:12]**
PPD[1:0] specifies the 802.1 priority to use for this packet. This field must be set regardless of whether or not the packet is already tagged (the TP bit is 1). Software must ensure the PPD[1:0] field is the same in this register as it is in the packet. Software is not permitted to use the null VLAN.

Note that when the CPU is forcing a port mask, the forwarding engine only uses the PPD[1:0] field to determine the output priority queue to use and it is not used to tag the packet. This is to allow the CPU to send exactly the packet it wants out the exact port priority queue it wants.

PVID[11:0] **VLAN ID** **[11:0]**

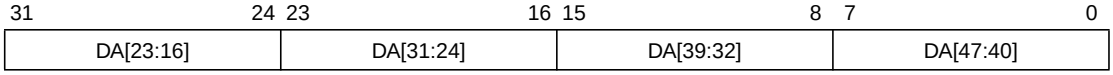
PVID[11:0] specifies the VLAN to use for this packet. This field must be set regardless of whether or not the packet is already tagged (the TP bit is 1). If the packet is already tagged, software must ensure the PVID[11:0] field is the same in this register as it is in the packet, and the PVID[11:0] field must also be the same as the PVID field in the buffer space before the packet. Software is not permitted to use the null VLAN.

Note that when the CPU is forcing a port mask, the forwarding engine only makes use of the PVID[11:0] field for the VLAN mirroring function and the field is not used for VLAN filtering or VLAN tagging. This is to allow the CPU to send exactly the packet it wants out the exact ports it wants.

Register: 0x3425.0024

PKT_TX_4 (Transmit Descriptor Fourth Word)
CPU:R/W
Default:0xxxxx.xxxx

This register contains the Transmit Descriptor Fourth Word for CPU packet transmission.



DA[23:16] **Dest Addr [23:16]** **[31:24]**

DA[31:24] **Dest Addr [31:24]** **[23:16]**

DA [39:32] **Dest Addr [39:32]** **[15:8]**

DA[47:40] **Dest Addr [47:40]** **[7:0]**

The CPU must write a copy of the Destination MAC Address (DA) to these locations. These fields must match the DA in the packet buffer. The forwarding engine uses the fields to perform address mirroring. When the OM bit is 0, these fields also determine the output port mask and the trunk link to use for the packet. The DA fields are organized such that a little endian processor can read the addresses directly as they are stored in memory, and copy them into the descriptor without byte swapping. Learning and aging is never performed on packets originating from the CPU.

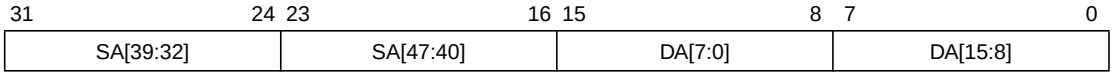
Note: The forwarding engine has special behavior when SA = DA. If SA = DA, the packet is *never* sent out any ports other than the mirror port. A packet with SA = DA can only be given to the CPU:

- if the CPU bit is set in the address table
- if it is associated with a VLAN whose CPU bit is set in the VLAN table
- if it is from a port being mirrored to the CPU
- if it is an extended RMON sample.

Register: 0x3425.0028

PKT_TX_5 (Transmit Descriptor Fifth Word)
CPU:R/W
Default:0xxxxx.xxxx

This register contains the Transmit Descriptor Fifth Word for CPU packet transmission.



SA[39:32] **Source Addr [39:32]** **[31:24]**

SA[47:40] **Source Addr [47:40]** **[23:16]**

DA[7:0] **Dest Addr [7:0]** **[15:8]**

DA[15:8] **Dest Addr [15:8]** **[7:0]**

A copy of the Destination MAC Address (DA) and Source MAC Address (SA) must be written by the CPU to these locations. These fields must match the DA and SA in the packet buffer. The forwarding engine uses these fields to perform address mirroring. When the OM bit = 0, these fields are also used to determine the output port mask and the trunk link to use for the packet. The DA and SA fields are organized such that a little endian processor can read the addresses directly as they are stored in memory, and copy them into the descriptor without byte swapping. Learning and aging is never performed on packets originating from the CPU.

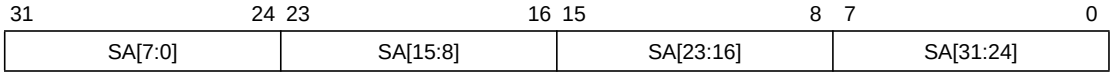
Note: The forwarding engine has special behavior when SA = DA. If SA = DA, the packet is *never* sent out any ports other than the mirror port. A packet with SA = DA can only be given to the CPU:

- if the CPU bit is set in the address table
- if it is associated with a VLAN whose CPU bit is set in the VLAN table
- if it is from a port being mirrored to the CPU
- if it is an extended RMON sample.

Register: 0x3425.002C

PKT_TX_6 (Transmit Descriptor Sixth Word)
CPU:R/W
Default:0xxxxx.xxxx

This register contains the Transmit Descriptor Sixth Word for CPU packet transmission.



SA[7:0] **Source Addr [7:0]** **[31:24]**

SA[15:8] **Source Addr [15:8]** **[23:16]**

SA [23:16] **Source Addr [23:16]** **[15:0]**

SA [31:24] **Source Addr [31:24]** **[15:0]**

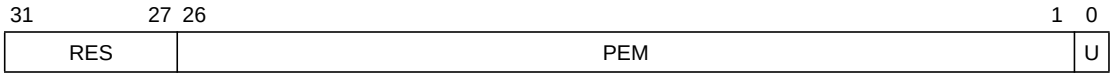
The CPU must write a copy of the Source MAC Address (SA) to these locations. These fields must match the SA in the packet buffer. The forwarding engine uses the fields to perform address mirroring. When the OM bit is 0, these fields also determine the output port mask and the trunk link to use for the packet. The SA fields are organized such that a little endian processor can read the addresses directly as they are stored in memory, and copy them into the descriptor without byte swapping. Learning and aging is never performed on packets originating from the CPU.

Note: The forwarding engine has special behavior when SA = DA. If SA = DA, the packet is *never* sent out any ports other than the mirror port. A packet with SA = DA can only be given to the CPU:

- if the CPU bit is set in the address table
- if it is associated with a VLAN whose CPU bit is set in the VLAN table
- if it is from a port being mirrored to the CPU
- if it is an extended RMON sample.

Register: 0x3425.0030

ERR_EN_MASK (Error Enable Mask)
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:27]**
These bits are reserved.

PEM[25:0] **Port Error Mask** **[26:1]**
For the CPU to receive an error packet (a packet with a CRC error or an oversize error), the following conditions must be met:

1) There is a register bit for each port to enable error packet reception. This must be set for the port in question (see the RCV_ERR register at 0x3420.0028)

2) The bit corresponding to that port must be set in the ERR_EN_MASK register at 0x3425.0030.

3) The ARL should find either the Destination Address (DA) or the Source Address (SA) in the ARL table AND the CPU bit in the entry must be set

or

The VLAN is enabled and the CB bit is set for this packet (see bit 52 in the subsection entitled "VLAN_TABLE_ENTRY (VLAN Table Entry Registers)" on page 8-295

or

Mirroring is enabled and the PM bit is set (see bit 0 in the subsection entitled "PORT_MIRR_MASK (Port Mirror Mask)" on page 8-263.

This same logic can be written symbolically (in C style) as follows:

The CPU will receive an error packet from port “x” if:

(CPU_ERR_ENx == 1) &&

(RCV_ERR_ENx == 1) &&

((CPU_dest == 1) || (CPU_source ==) ||

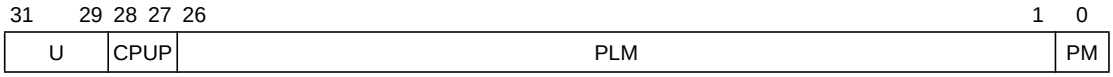
(CPU_vlan == 1) || (CPU_portmirr == 1))

U **Unused** **0**

This bit is not used.

Register: 0x3425.0034

SRC_PORT_LOCK (Source Port Locking)
CPU:R/W
Default:0x0000.0000



U **Unused** **[31:29]**
These bits are not used.

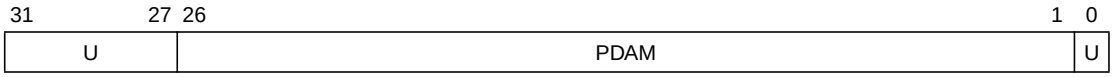
CPUP[1:0] **CPU Priority** **[28:27]**
When the PM bit is set, the violating packet is forwarded to the CPU and is placed in one of four CPU receive priority queues depending on CPUP[1:0]. A value of 0b00 selects the lowest priority queue and a value of 0b11 selects the highest priorIty queue.

PLM[25:0] **Port Lock Mask** **[26:1]**
When a given bit position (corresponding to a port) is set, address moves, learns, and aging are not allowed for packets sourced from that port. Only traffic that was previously learned or programmed as being sourced from the port, as specified in the source port field of each address table entry, is forwarded. Packets that are from a locked port that result in a source port mismatch or that are not found in the address table are filtered or sent to the CPU. Setting a PLM bit for a port also prevents the aging logic from aging entries on the port. The Don't Age bit is considered set when the Port Lock Mask bit is set. Packets containing unknown destination addresses are also not forwarded to a locked port. When the PLM bit is set, the Age bit for all address table entries with the corresponding source port are redefined to become the Moves bit. This bit determines if an entry may be relearned on another port. Moves are allowed to other source locked ports if the Age/Moves bit is set.

PM **CPU Source Port Mismatch** **0**
If this bit is set, all packets that result in a mismatch are forwarded to the CPU only. If a mismatch is encountered and the bit is 0, the packet is discarded if not sent to the mirror port.

Register: 0x3425.0038

PORT_AGE_CTRL (Port Aging Control)
CPU:R/W
Default:0x0000.0000



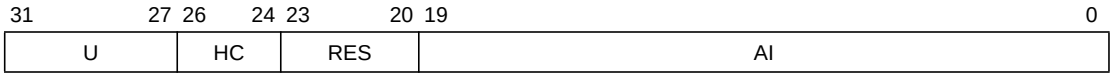
U **Unused** **[31:27]**
These bits are not used.

PDAM[25:0] **Port Don't Age Mask** **[26:1]**
When a port bit in PDAM[25:0] is 0, aging is performed normally for address table entries with the corresponding source port. When a port bit is set, all source addresses received from the corresponding port are not aged. When the Don't Age bit is set, it overrides the setting of the Don't Age bit in the Address table for entries with the corresponding source port. When set to 1, the Age bit for all address table entries with the corresponding source port are redefined to become the Moves bit. This bit determines if an entry may be relearned on another port.

U **Unused** **0**
This bit is not used.

Register: 0x3425.003C

AGE_INTERVAL (Aging Interval)
CPU:R/W
Default:0xx6xx.003C



R **Reserved** **[31:27]**
These bits are reserved.

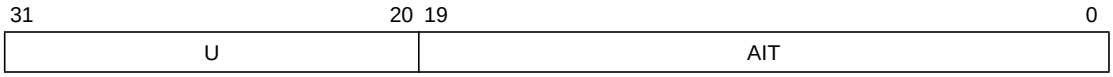
HC[2:0] **Hash Count** **[26:24]**
This field sets the number of Hash functions the switch uses to search for entries in the address table. By default, six unique hash functions are applied to each source and destination address. The value of the field must be between 1 and 6.

RES **Reserved** **[23:20]**
These bits are reserved.

AI[19:0] **Age Interval** **[19:0]**
The AI[19:0] field is multiplied by 5 seconds to form the physical Aging Interval. When the Age Interval expires, the Age Interval is reloaded and the Aging logic parses the entire address table updating and Aging entries. The physical interval has a range from 0 to 1,000,000 seconds. A value of zero globally disables aging. The range of the AI[19:0] field is from 0 to 0xFFFFF.

Register: 0x3425.0040

AGE_TIMER (Aging Interval Timer)
CPU:Read Only
Default:0xxxx0.0000

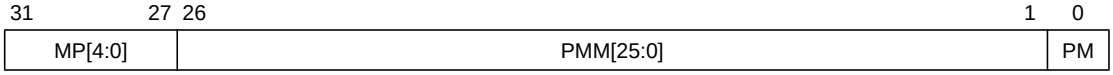


U **Unused** **[31:20]**
These bits are not used.

AIT **Age Interval Timer Count** **[19:0]**
This field contains the Age Interval timer count.

Register: 0x3425.0044

PORT_MIRR_MASK (Port Mirror Mask)
CPU:R/W
Default:0x0000.0000



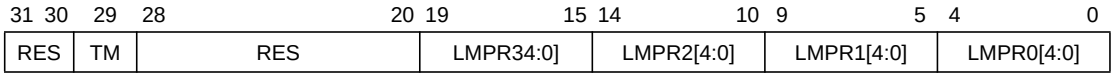
MP[4:0] **Mirror Port** **[31:27]**
This field is loaded with the port that should receive the mirrored traffic. The range of port values is 1 through 26. A value of 0 disables this function. Software must never program the mirror port to the STP blocking or learning states.

PMM[25:0] **Port Mirror Mask** **[26:1]**
When a port bit in this field is set, all traffic sent or received on the corresponding port is forwarded to the port indicated in the Mirror Port field. The bit corresponding to the port selected as the Mirror Port is redefined to control general flooding to the Mirror Port. If the corresponding mask bit is set, then packets are not flooded out the mirror port. Note that packets flooded to another port that has its Port Mirror Mask bit set are flooded to the mirror port.

PM **CPU Port Mirror** **0**
When the PM bit is set, the traffic is sent to the CPU as well as the Mirror Port. This traffic is always queued to the lowest priority receive queue(CPUP[1:0] = 0b00).

Register: 0x3425.0048

LMPR (Logical Map Physical)
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:30]**
These bits are reserved.

TM **Trunk Method** **29**
1 = SA only Trunking
0 = SA/DA Trunking

RES **Reserved** **[28:20]**
These bits are reserved.

LMPR3[4:0] **Logical Port 3 Mapping** **[19:15]**
The LMPR3[4:0] field sets the physical port for logical trunk link 3. A value of 0b0000 removes the logical port from the trunk. Valid values are from 0 to 26.

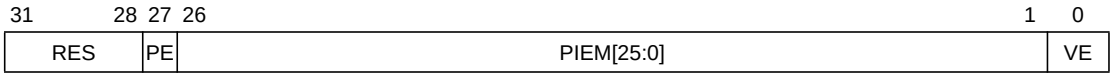
LMPR2[4:0] **Logical Port 2 Mapping** **[14:10]**
The LMPR2[4:0] field sets the physical port for logical trunk link 2. A value of 0b0000 removes the logical port from the trunk. Valid values are from 0 to 26.

LMPR1[4:0] **Logical Port 1 Mapping** **[9:5]**
The LMPR1[4:0] field sets the physical port for logical trunk link 1. A value of 0b0000 removes the logical port from the trunk. Valid values are from 0 to 26.

LMPR0[4:0] **Logical Port 0 Mapping** **[4:0]**
The LMPR0[4:0] field sets the physical port for logical trunk link 0. A value of 0b0000 removes the logical port from the trunk. Valid values are from 0 to 26.

Register: 0x3425.004C

VLAN_ING_ENABLE (VLAN Ingress Enable)
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:28]**
These bits are reserved.

PE **Priority Enable** **27**
This bit only has a function when the VE bit is 0. When VE is set, priority functions are enabled by default. When VE is 0, this bit determines if the priority function is enabled.

PIEM[25:0] **Port Ingress Enable Mask** **[26:1]**
Each PIEM port bit controls whether Ingress filtering will occur. By default, Ingress filtering is disabled.

VE **VLAN Enable** **0**
This bit is the global VLAN enable bit. By default, VLANs are disabled in the switch. When disabled, the switch does not use the VLAN or priority fields in the forwarding decision. When VE is set, the VLAN and priority functions described in this section are enabled.

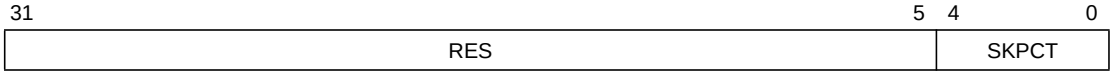
Register: 0x3425.0050–0x3425.0057

RESERVED
CPU:-
Default:0xxxxx.xxxx

RES **Reserved** **[31:0]**
These bits are reserved.

Register: 0x3425.0058

RMON_SKIPCNT_1 (RMON Skip Count Port 1)
CPU:R/W
Default:0xxxxx.xx00

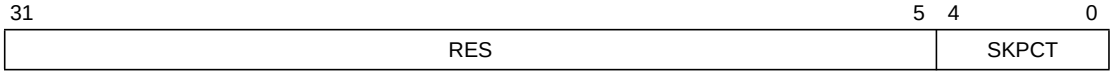


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
SKPCNT[4:0] is the countdown register that is being decremented for each packet received from or queued for transmission out on the link. When a packet causes this register to decrement from 1 to 0, that packet is sampled. No more samples are taken for this port until the CPU writes a new value into SKPCNT[4:0] and it is decremented to 0 again. This counter decrements only when extended RMON is enabled. The CPU may write to this register at any time.

Register: 0x3425.005C

RMON_SKIPCNT_2 (RMON Skip Count Port 2)
CPU:R/W
Default:0xxxxx.xx00

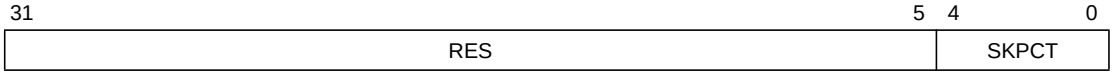


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0060

RMON_SKIPCNT_3 (RMON Skip Count Port 3)
CPU:R/W
Default:0xxxxx.xx00

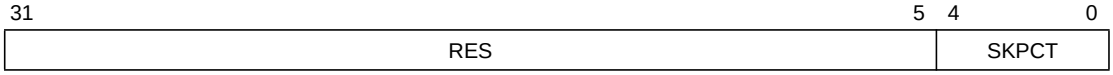


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0064

RMON_SKIPCNT_4 (RMON Skip Count Port 4)
CPU:R/W
Default:0xxxxx.xx00

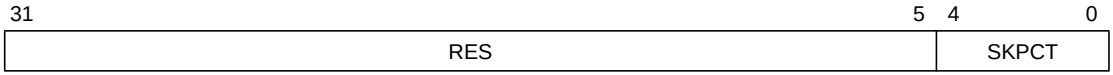


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: **0x3425.0068**

RMON_SKIPCNT_5 (RMON Skip Count Port 5)
CPU:R/W
Default:0xxxxx.xx00

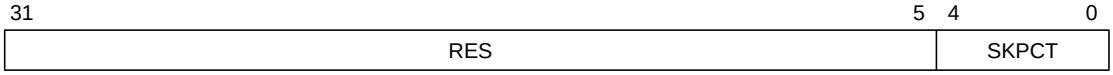


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.006C

RMON_SKIPCNT_6 (RMON Skip Count Port 6)
CPU:R/W
Default:0xxxxx.xx00

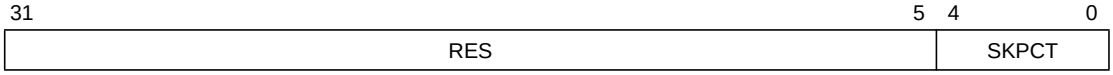


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0070

RMON_SKIPCNT_7 (RMON Skip Count Port 7)
CPU:R/W
Default:0xxxxx.xx00

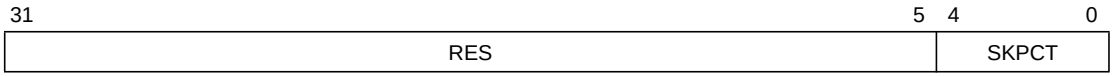


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0074

RMON_SKIPCNT_8 (RMON Skip Count Port 8)
CPU:R/W
Default:0xxxxx.xx00

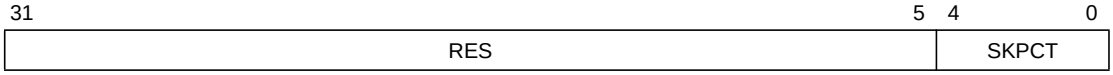


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0078

RMON_SKIPCNT_9 (RMON Skip Count Port 9)
CPU:R/W
Default:0xxxxx.xx00

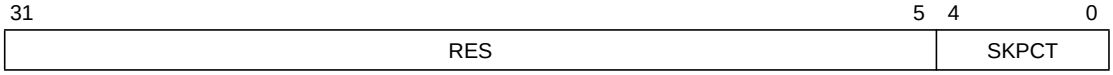


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.007C

RMON_SKIPCNT_10 (RMON Skip Count Port 10)
CPU:R/W
Default:0xxxxx.xx00

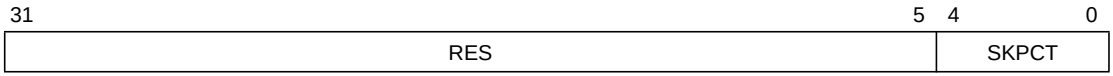


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0080

RMON_SKIPCNT_11 (RMON Skip Count Port 11)
CPU:R/W
Default:0xxxxx.xx00

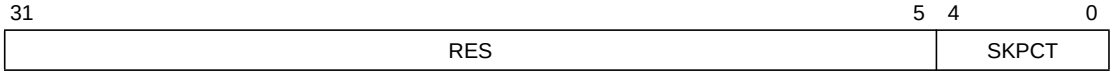


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0084

RMON_SKIPCNT_12 (RMON Skip Count Port 12)
CPU:R/W
Default:0xxxxx.xx00

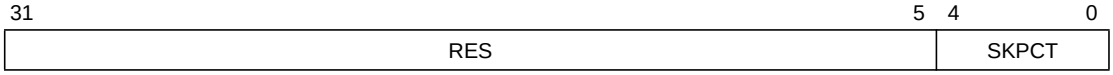


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0088

RMON_SKIPCNT_13 (RMON Skip Count Port 13)
CPU:R/W
Default:0xxxxx.xx00

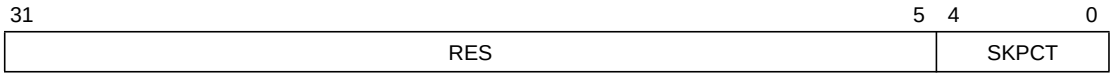


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.008C

RMON_SKIPCNT_14 (RMON Skip Count Port 14)
CPU:R/W
Default:0xxxxx.xx00

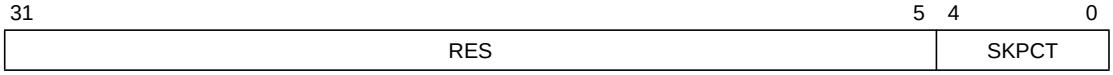


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0090

RMON_SKIPCNT_15 (RMON Skip Count Port 15)
CPU:R/W
Default:0xxxxx.xx00

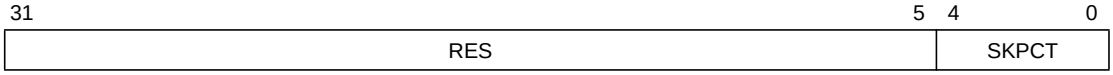


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0094

RMON_SKIPCNT_16 (RMON Skip Count Port 16)
CPU:R/W
Default:0xxxxx.xx00

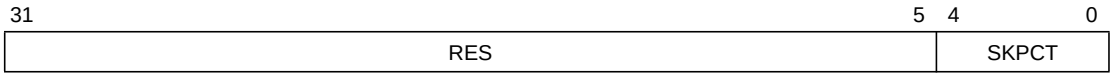


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0098

RMON_SKIPCNT_17 (RMON Skip Count Port 17)
CPU:R/W
Default:0xxxxx.xx00

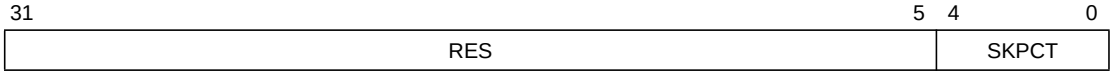


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.009C

RMON_SKIPCNT_18 (RMON Skip Count Port 18)
CPU:R/W
Default:0xxxxx.xx00

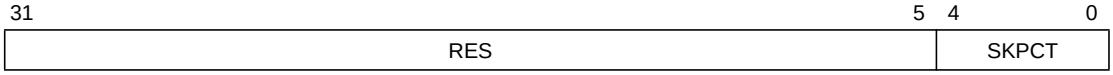


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.00A0

RMON_SKIPCNT_19 (RMON Skip Count Port 19)
CPU:R/W
Default:0xxxxx.xx00

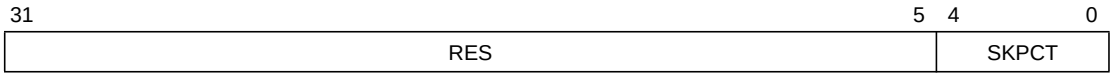


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.00A4

RMON_SKIPCNT_20 (RMON Skip Count Port 20)
CPU:R/W
Default:0xxxxx.xx00

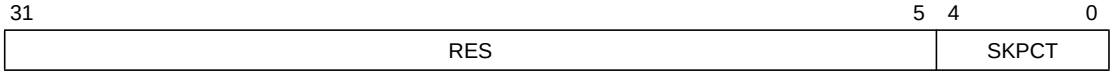


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.00A8

RMON_SKIPCNT_21 (RMON Skip Count Port 21)
CPU:R/W
Default:0xxxxx.xx00

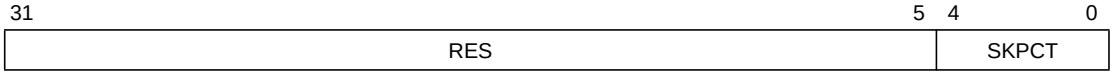


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.00AC

RMON_SKIPCNT_22 (RMON Skip Count Port 22)
CPU:R/W
Default:0xxxxx.xx00

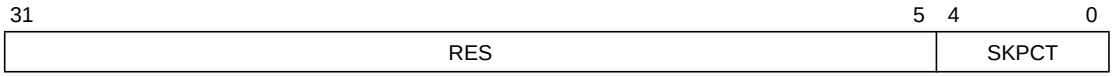


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.00B0

RMON_SKIPCNT_23 (RMON Skip Count Port 23)
CPU:R/W
Default:0xxxxx.xx00

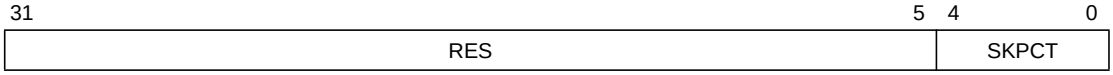


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.00B4

RMON_SKIPCNT_24 (RMON Skip Count Port 24)
CPU:R/W
Default:0xxxxx.xx00

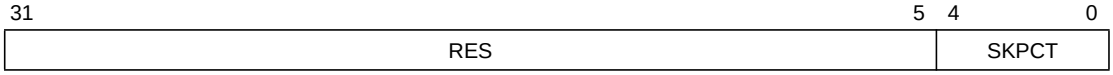


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.00B8

RMON_SKIPCNT_25 (RMON Skip Count Port 25)
CPU:R/W
Default:0xxxxx.xx00

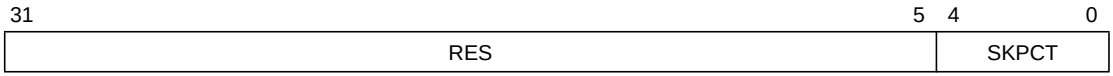


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.00BC

RMON_SKIPCNT_26 (RMON Skip Count Port 26)
CPU:R/W
Default:0xxxxx.xx00

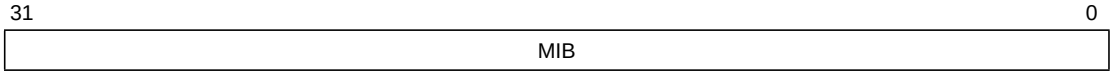


R **Reserved** **[31:5]**
These bits are reserved.

SKPCT[4:0] **Skip Count** **[4:0]**
See the description in register 0x3425.0058.

Register: 0x3425.0190

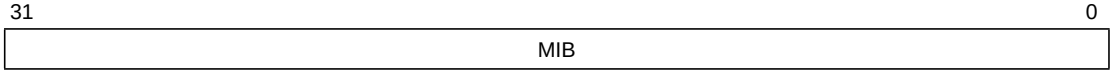
IF_OUT_DISCARDS_1 (Output Packet Discards, Port 1)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 1** **[31:0]**
This field is the switch engine OutDrops counter. It shows the number of packets that were discarded on this output port due to insufficient transmit descriptors for this output port.

Register: 0x3425.0194

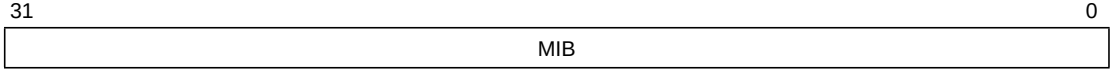
IF_OUT_DISCARDS_2 (Output Packet Discards, Port 2)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 2** **[31:0]**
See port 1 description.

Register: 0x3425.0198

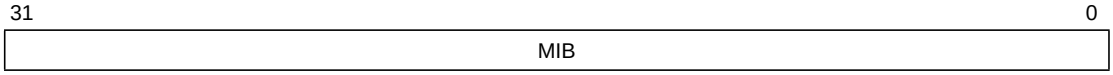
IF_OUT_DISCARDS_3 (Output Packet Discards, Port 3)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 3** **[31:0]**
See port 1 description.

Register: 0x3425.019C

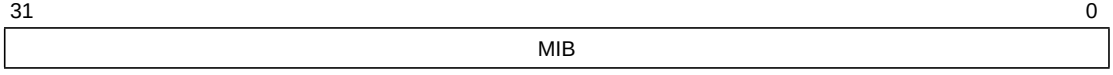
IF_OUT_DISCARDS_4 (Output Packet Discards, Port 4)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 4** See port 1 description.	**[31:0]**

Register: 0x3425.01A0

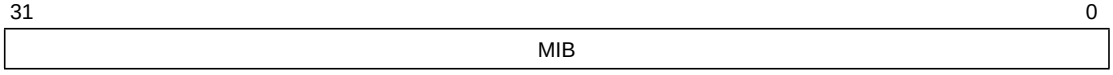
IF_OUT_DISCARDS_5 (Output Packet Discards, Port 5)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 5** See port 1 description.	**[31:0]**

Register: 0x3425.01A4

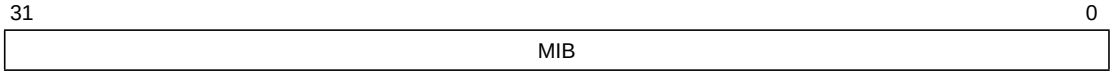
IF_OUT_DISCARDS_6 (Output Packet Discards, Port 6)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 6** See port 1 description.	**[31:0]**

Register: 0x3425.01A8

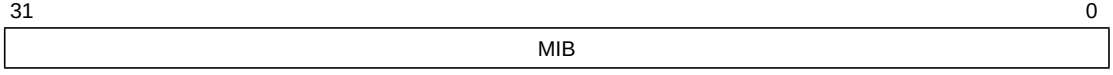
IF_OUT_DISCARDS_7 (Output Packet Discards, Port 7)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 7** **[31:0]**
See port 1 description.

Register: 0x3425.01AC

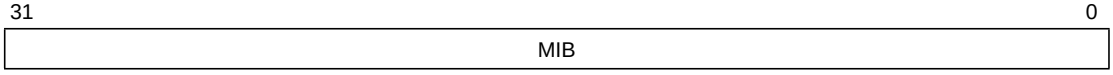
IF_OUT_DISCARDS_8 (Output Packet Discards, Port 8)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 8** **[31:0]**
See port 1 description.

Register: 0x3425.01B0

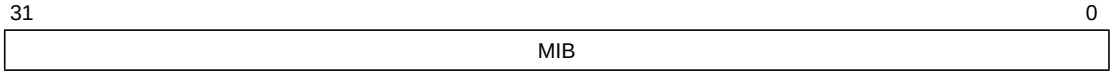
IF_OUT_DISCARDS_9 (Output Packet Discards, Port 9)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 9** **[31:0]**
See port 1 description.

Register: 0x3425.01B4

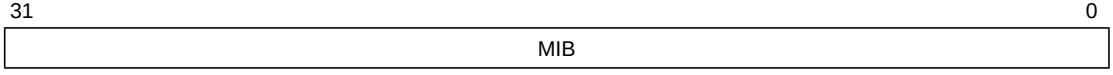
IF_OUT_DISCARDS_10 (Output Packet Discards, Port 10)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 10** See port 1 description.	**[31:0]**

Register: 0x3425.01B8

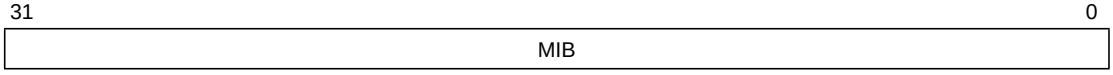
IF_OUT_DISCARDS_11 (Output Packet Discards, Port 11)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 11** See port 1 description.	**[31:0]**

Register: 0x3425.01BC

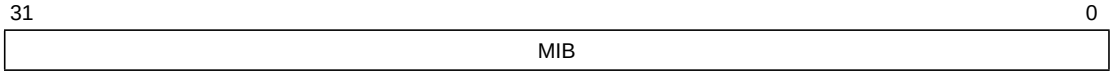
IF_OUT_DISCARDS_12 (Output Packet Discards, Port 12)
CPU:Read Clear/Read Only
Default:0x0000 0000



MIB	**MIB ifOutDiscards Counter for Port 12** See port 1 description.	**[31:0]**

Register: 0x3425.01C0

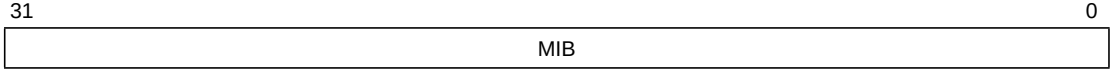
IF_OUT_DISCARDS_13 (Output Packet Discards, Port 13)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 13** **[31:0]**
See port 1 description.

Register: 0x3425.01C4

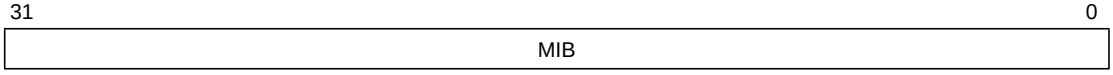
IF_OUT_DISCARDS_14 (Output Packet Discards, Port 14)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 14** **[31:0]**
See port 1 description.

Register: 0x3425.01C8

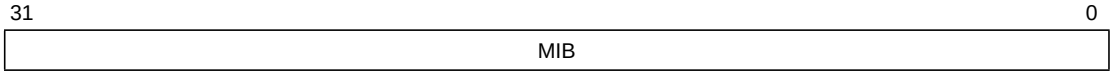
IF_OUT_DISCARDS_15 (Output Packet Discards, Port 15)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 15** **[31:0]**
See port 1 description.

Register: 0x3425.01CC

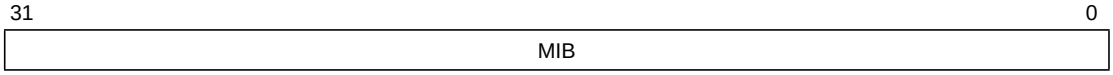
IF_OUT_DISCARDS_16 (Output Packet Discards, Port 16)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 16** See port 1 description.	**[31:0]**

Register: 0x3425.01D0

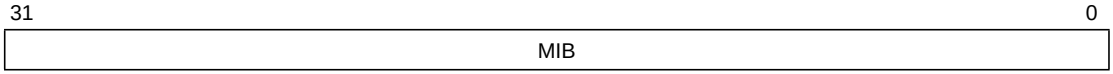
IF_OUT_DISCARDS_17 (Output Packet Discards, Port 17)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 17** See port 1 description.	**[31:0]**

Register: 0x3425.01D4

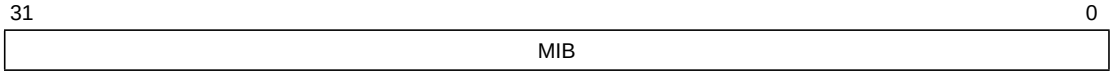
IF_OUT_DISCARDS_18 (Output Packet Discards, Port 18)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 18** See port 1 description.	**[31:0]**

Register: 0x3425.01D8

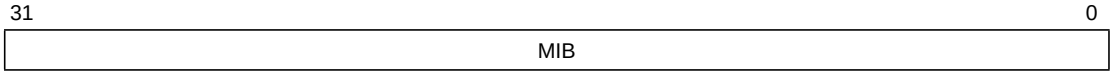
IF_OUT_DISCARDS_19 (Output Packet Discards, Port 19)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 19** **[31:0]**
See port 1 description.

Register: 0x3425.01DC

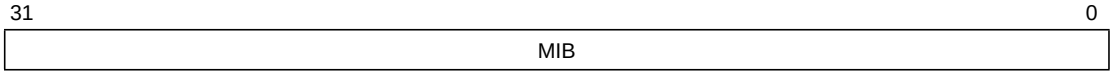
IF_OUT_DISCARDS_20 (Output Packet Discards, Port 20)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 20** **[31:0]**
See port 1 description.

Register: 0x3425.01E0

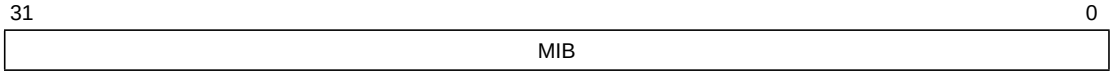
IF_OUT_DISCARDS_21 (Output Packet Discards, Port 21)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB ifOutDiscards Counter for Port 21** **[31:0]**
See port 1 description.

Register: 0x3425.01E4

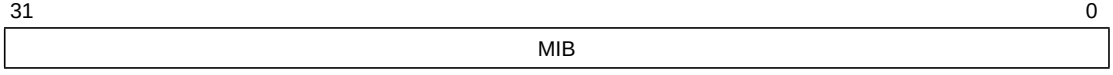
IF_OUT_DISCARDS_22 (Output Packet Discards, Port 22)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 22** See port 1 description.	**[31:0]**

Register: 0x3425.01E8

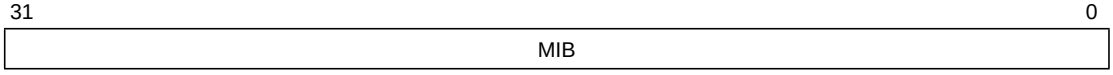
IF_OUT_DISCARDS_23 (Output Packet Discards, Port 23)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 23** See port 1 description.	**[31:0]**

Register: 0x3425.01EC

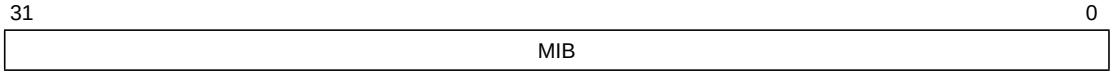
IF_OUT_DISCARDS_24 (Output Packet Discards, Port 24)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 24** See port 1 description.	**[31:0]**

Register: 0x3425.01F0

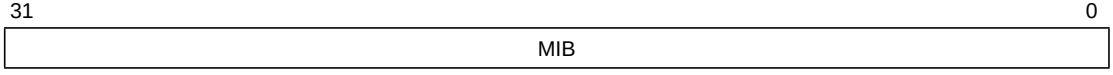
IF_OUT_DISCARDS_25 (Output Packet Discards, Port 25)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 25** See port 1 description.	**[31:0]**

Register: 0x3425.01F4

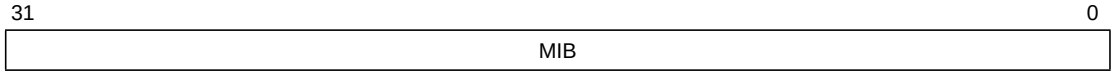
IF_OUT_DISCARDS_26 (Output Packet Discards, Port 26)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB	**MIB ifOutDiscards Counter for Port 26** See port 1 description.	**[31:0]**

Register: 0x3425.01F8

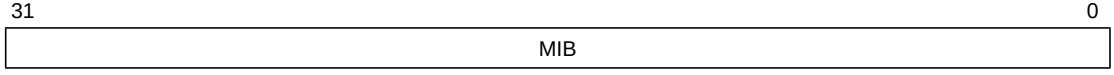
PORT_IN_DISCARDS_1 (Input Packet Port Discards, Port 1)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 1 [31:0]**
This field is a count of frames filtered. It specifies the number of error-free packets received and able to be buffered successfully, but which were not forwarded to any other ports of the switch, including the CPU port (a packet for which the CPU bit is set during the destination address lookup does not cause this counter to increment). Port mirroring, VLAN mirroring, and extended RMON sampling are not considered forwarding, and thus these types of packets cause this counter to increment if they are not forwarded by the switch.

Register: 0x3425.01FC

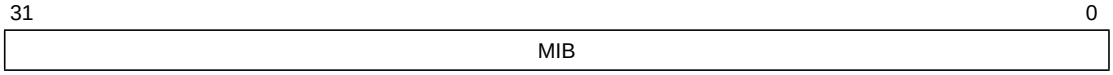
PORT_IN_DISCARDS_2 (Input Packet Port Discards, Port 2)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 2 [31:0]**
See port 1 description.

Register: 0x3425.0200

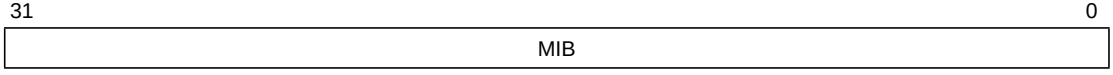
PORT_IN_DISCARDS_3 (Input Packet Port Discards, Port 3)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 3** **[31:0]**
See port 1 description.

Register. 0x3425.0204

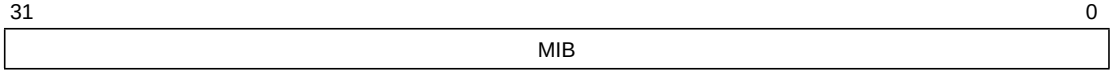
PORT_IN_DISCARDS_4 (Input Packet Port Discards, Port 4)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 4** **[31:0]**
See port 1 description.

Register: 0x3425.0208

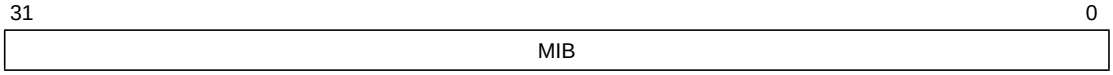
PORT_IN_DISCARDS_5 (Input Packet Port Discards, Port 5)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 5** **[31:0]**
See port 1 description.

Register. 0x3425.020C

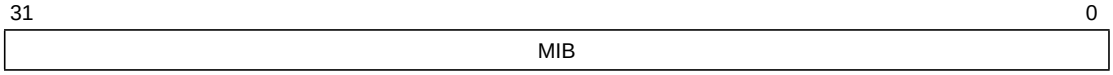
PORT_IN_DISCARDS_6 (Input Packet Port Discards, Port 6)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 6** **[31:0]**
See port 1 description.

Register: 0x3425.0210

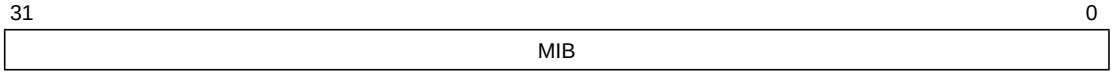
PORT_IN_DISCARDS_7 (Input Packet Port Discards, Port 7)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 7** **[31:0]**
See port 1 description.

Register. 0x3425.0214

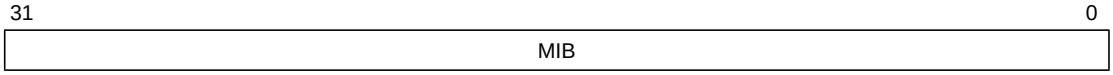
PORT_IN_DISCARDS_8 (Input Packet Port Discards, Port 8)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 8** **[31:0]**
See port 1 description.

Register: 0x3425.0218

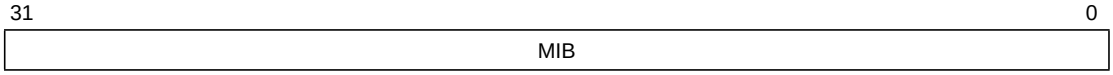
PORT_IN_DISCARDS_9 (Input Packet Port Discards, Port 9)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 9** **[31:0]**
See port 1 description.

Register. 0x3425.021C

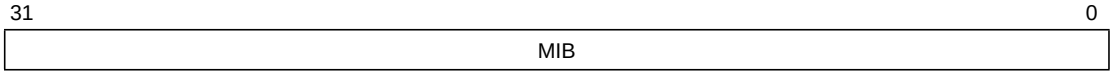
PORT_IN_DISCARDS_10 (Input Packet Port Discards, Port 10)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 10** **[31:0]**
See port 1 description.

Register: 0x3425.0220

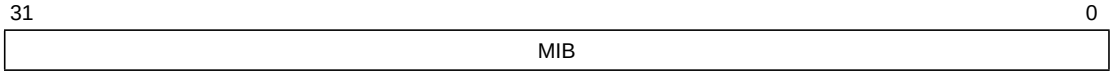
PORT_IN_DISCARDS_11 (Input Packet Port Discards, Port 11)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 11** **[31:0]**
See port 1 description.

Register. 0x3425.0224

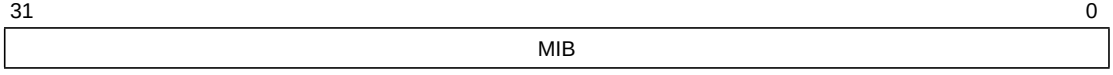
PORT_IN_DISCARDS_12 (Input Packet Port Discards, Port 12)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 12 [31:0]**
See port 1 description.

Register: 0x3425.0228

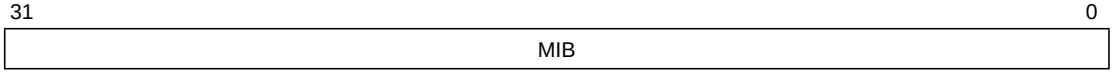
PORT_IN_DISCARDS_13 (Input Packet Port Discards, Port 13)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 13 [31:0]**
See port 1 description.

Register. 0x3425.022C

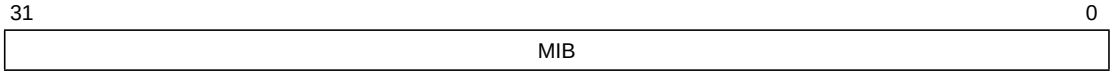
PORT_IN_DISCARDS_14 (Input Packet Port Discards, Port 14)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 14 [31:0]**
See port 1 description.

Register: 0x3425.0230

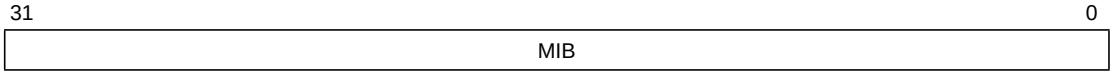
PORT_IN_DISCARDS_15 (Input Packet Port Discards, Port 15)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 15 [31:0]**
See port 1 description.

Register. 0x3425.0234

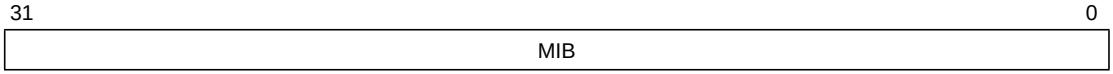
PORT_IN_DISCARDS_16 (Input Packet Port Discards, Port 16)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 16 [31:0]**
See port 1 description.

Register: 0x3425.0238

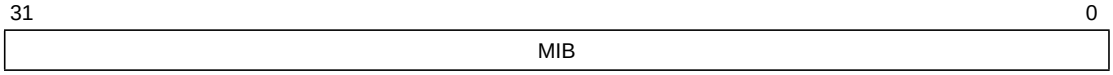
PORT_IN_DISCARDS_17 (Input Packet Port Discards, Port 17)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 17 [31:0]**
See port 1 description.

Register. 0x3425.023C

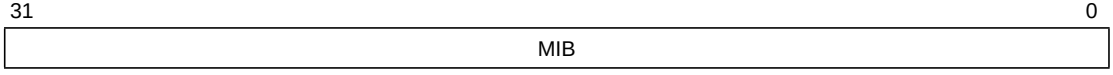
PORT_IN_DISCARDS_18 (Input Packet Port Discards, Port 18)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 18 [31:0]**
See port 1 description.

Register: 0x3425.0240

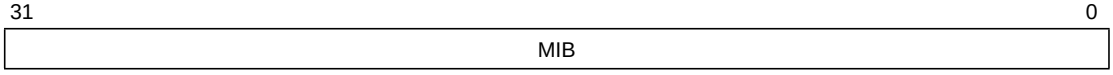
PORT_IN_DISCARDS_19 (Input Packet Port Discards, Port 19)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 19 [31:0]**
See port 1 description.

Register. 00x3425.0244

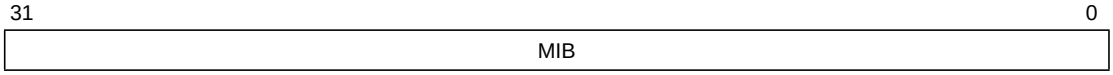
PORT_IN_DISCARDS_20 (Input Packet Port Discards, Port 20)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 20 [31:0]**
See port 1 description.

Register: 0x3425.0248

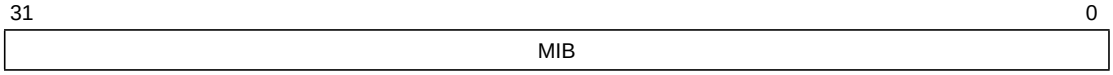
PORT_IN_DISCARDS_21 (Input Packet Port Discards, Port 21)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 21 [31:0]**
See port 1 description.

Register. 0x3425.024C

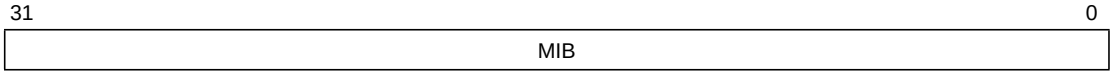
PORT_IN_DISCARDS_22 (Input Packet Port Discards, Port 22)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 22 [31:0]**
See port 1 description.

Register: 0x3425.0250

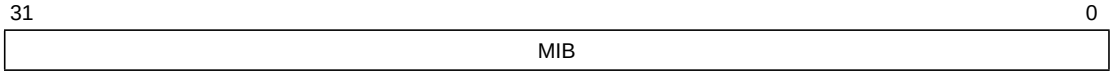
PORT_IN_DISCARDS_23 (Input Packet Port Discards, Port 23)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 23 [31:0]**
See port 1 description.

Register. 0x3425.0254

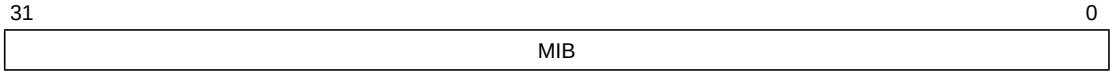
PORT_IN_DISCARDS_24 (Input Packet Port Discards, Port 24)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 24 [31:0]**
See port 1 description.

Register: 0x3425.0258

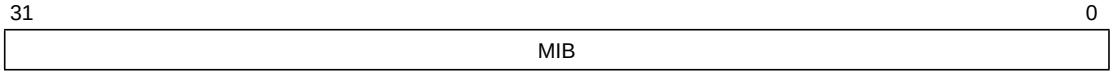
PORT_IN_DISCARDS_25 (Input Packet Port Discards, Port 25)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 25 [31:0]**
See port 1 description.

Register. 0x3425.025C

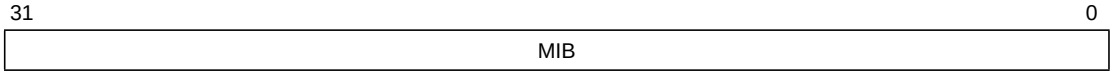
PORT_IN_DISCARDS_26 (Input Packet Port Discards, Port 26)
CPU:Read Clear/Read Only
Default:0x0000.0000



MIB **MIB dot1dTpPortInDiscards Counter Port 26 [31:0]**
See port 1 description.

Register: 0x3425.0260

LEARNED_ENTRY_DISCARDS
CPU:Read Clear/Read Only
Default:0x0000.0000

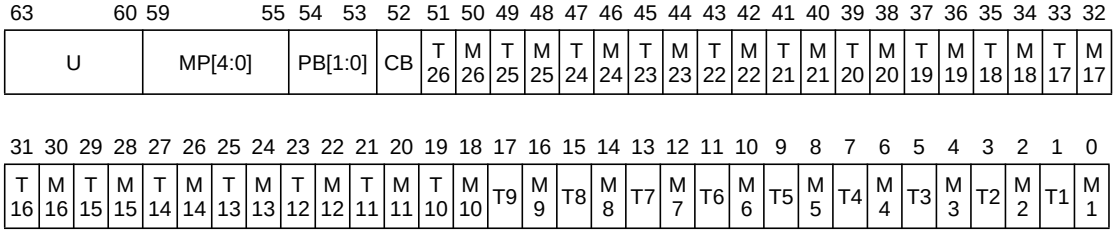


MIB **MIB dot1dTpLearned Entry Discards for the Switch** **[31:0]**

This field is a global counter that increments each time an address is expelled from the switch address table to make room for another entry that is being learned.

Register: 0x3425.0300–0x3425.03FF

VLAN_TABLE_ENTRY (VLAN Table Entry Registers)
CPU:R/W
Default:0xxxxx.xxxx



The VLAN table is formed from 32 registers, each 60 bits wide (the 4 MSBs are ignored). The registers are mapped in the address space from 0x3425.0000 to 0x3425.03FF.

Locations 0, 8, 10, 18 ... F8 are bits [31:0] of the VLAN_TABLE_ENTRY registers and location 4, C, 14, 1C ... FC are bits [63:32] of the VLAN_TABLE_ENTRY registers.

U **Unused** **[63:60]**
These bits are not used.

MP[4:0] **Mirror Port** **[59:55]**
The MP field is used to forward all packets on a given VLAN to a specified port, in addition to the intended port(s). The MP values can range from 1 to 26, with 0 indicating no port mirror.

This scheme allows multiple VLANs to be monitored on separate ports. Also, the VLAN target Mirror Port can be assigned to a port other than that selected in the Port Mirror Mask Register. Software must never program mirror target ports to the STP blocking or learning states. When VLANs are enabled, every packet that arrives at a port that is not disabled is checked against the VLAN table to determine if the packet should be forwarded to the corresponding mirror port. So even if a port is in blocking or learning mode this check is performed.

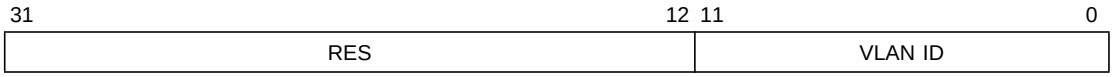
PB[1:0]	**CPU Priority Bit** When the CB is 1, the packet is also sent to the CPU and placed in the receive queue corresponding to the PB[1:0] field (lowest priority = 0b00; highest priority = 0b11).	**[54:53]**
CB	**CPU Copy Bit** When the CB bit is 1 for a given VLAN ID, packets with the VLAN ID are forwarded to the CPU as well as to its intended destination. The packet is placed in one of the four CPU receive queues as indicated in the CPU priority field. When VLANs are enabled, every packet that arrives at a port that is not disabled is checked against the VLAN table to determine if the packet should be forwarded to the CPU. So even if a port is in blocking or learning mode, this check is performed.	**52**
T26	**Tagged Bit of Port 26** The Tagged field describes how a tagged packet should be forwarded out a port that is a member of the VLAN. If the bit is 1, the packet is forwarded tagged, otherwise the packet is forwarded without a tag field.	**51**
M26	**Member Bit of Port 26** The Member bit, when 1, indicates that the given port is a member of the corresponding VLAN. Only ports that are a member of a given VLAN may forward traffic on that VLAN.	**50**
T25	**Tagged Bit of Port 25**	**49**
M25	**Member Bit of Port 25**	**48**
T24	**Tagged Bit of Port 24**	**47**
M24	**Member Bit of Port 24**	**46**
T23	**Tagged Bit of Port 23**	**45**
M23	**Member Bit of Port 23**	**44**
T22	**Tagged Bit of Port 22**	**43**
M22	**Member Bit of Port 22**	**42**
T21	**Tagged Bit of Port 21**	**41**
M21	**Member Bit of Port 21**	**40**

T20	**Tagged Bit of Port 20**	**39**
M20	**Member Bit of Port 20**	**38**
T19	**Tagged Bit of Port 19**	**37**
M19	**Member Bit of Port 19**	**36**
T18	**Tagged Bit of Port 18**	**35**
M18	**Member Bit of Port 18**	**34**
T17	**Tagged Bit of Port 17**	**33**
M17	**Member Bit of Port 17**	**32**
T16	**Tagged Bit of Port 16**	**31**
M16	**Member Bit of Port 16**	**30**
T15	**Tagged Bit of Port 15**	**29**
M15	**Member Bit of Port 15**	**28**
T14	**Tagged Bit of Port 14**	**27**
M14	**Member Bit of Port 14**	**26**
T13	**Tagged Bit of Port 13**	**25**
M13	**Member Bit of Port 13**	**24**
T12	**Tagged Bit of Port 12**	**23**
M12	**Member Bit of Port 12**	**22**
T11	**Tagged Bit of Port 11**	**21**
M11	**Member Bit of Port 11**	**20**
T10	**Tagged Bit of Port 10**	**19**
M10	**Member Bit of Port 10**	**18**
T9	**Tagged Bit of Port 9**	**17**
M9	**Member Bit of Port**	**16**
T8	**Tagged Bit of Port 8**	**15**

M8	**Member Bit of Port 8**	**14**
T7	**Tagged Bit of Port 7**	**13**
M7	**Member Bit of Port 7**	**12**
T6	**Tagged Bit of Port 6**	**11**
M6	**Member Bit of Port 6**	**10**
T5	**Tagged Bit of Port 5**	**9**
M5	**Member Bit of Port 5**	**8**
T4	**Tagged Bit of Port 4**	**7**
M4	**Member Bit of Port 4**	**6**
T3	**Tagged Bit of Port 3**	**5**
M3	**Member Bit of Port 3**	**4**
T2	**Tagged Bit of Port 2**	**3**
M2	**Member Bit of Port 2**	**2**
T1	**Tagged Bit of Port 1**	**1**
M1	**Member Bit of Port 1**	**0**

Register: 0x3425.0400–0x3425.047F

VLAN_CAM (VLAN CAM Registers)
CPU:R/W
Default:0xxxxx.xxxx



There are 32 12-bit VLAN CAM registers that store the VLAN IDs. They are mapped into the 0x3425.0400 to 0x3425.047F address space.

RES **Reserved** **[31:12]**
These bits are reserved.

VLAN ID[11:0] **VLAN ID** **[11:0]**
The VLAN ID[11:0] field contains the VLAN ID corresponding to each of the VLAN table entries.

The VLAN_CAM with VLAN_TABLE_ENTRY works as Content Addressable Memory. When making a forwarding decision, the switch engine compares the VLAN_ID extracted by the port to the VLAN_ID contained in the VLAN_CAM registers. If there is a match, an index into the VLAN table is returned. This index selects a corresponding 60-bit entry. If the CAM access fails to locate the VLAN ID, a VLAN port mask of all zeroes is used, indicating the packet should be dropped.

Register: 0x3425.8000–0x3425.FFFF

ARL_TABLE
CPU:R/W
Default:0xxxxx.xxxx

	63 60	59 47	46 32
Unicast Word	U	MACA	
Multicast 2 Words	U	MACA	
	U	PM	

	31 21	20 13	12 11	7	6 5	4	3	2	1	0
Unicast Word	MACA		SP		P	CPU	PM	E	DA	A
Multicast 2 Words	MACA		VLAN ID							
	PM	U	PO	MT		PC	EMP	PM	U	

There are 4096 60-bit ARL Table entries in this address space.

Note: The CPU should never directly access these registers. The only access to the registers should be through the ARL command interface.

The even locations (0, 2, 4, 6, 8, A, C, E...) are bits [31:0] of a table entry, and the odd locations (1, 3, 5, 7, 9, B, D, F...) contain bits [59:32] of the Table Entry. The upper four bits of these locations are not used.

Locations {1,0}, {3,2}, {5,4}, {7,6}... can carry a Unicast Entry, while locations {3,2,1,0}, {7,6,5,4}, {B,A,9,8}... can carry a Multicast Entry.

Unicast Entries

Unicast Entries use one 64-bit word, which occupies two 32-bit locations.

MACA[47:0] **MAC Address** **[59:12]**
This is the 48-bit unicast address. Bit 52 is always zero for unicast entries.

SP[4:0] **Source Port** **[11:7]**
This field specifies the port on which the address was last sourced. Valid entries are 0 to 26, where port 0 is the CPU. If SP[4:0] is 0, the packet is sent to the CPU and

placed in one of the four queues as determined from the P[1:0] field.

P[1:0] **Priority CPU** **[6:5]**

If the packet is to be copied to the CPU, as determined by the CPU bit or a source port of 0, then this two-bit field determines which of the four CPU receive queues the packet uses, as shown in the table.

P[1:0]	CPU Receive Queue
0b00	Lowest
0b01	Low
0b10	High
0b11	Highest

CPU **CPU** **4**

If this bit is set, the packet is also sent to one of the CPU four priority queues, in addition to the intended destination. Only the CPU sets this bit. By default this bit is 0. Setting the Source Port (SP4:0]) field to zero also causes the packet to be forwarded to the CPU. The CPU queue is selected through P[1:0] field.

PM **Port Mirror** **3**

If this bit is set, the packet is also sent to the port specified in the CPU Port Mirror (PM) field of the Port Mirror Mask Register (0x3425.0044), in addition to the intended destination. Only the CPU sets this bit. By default, this bit is 0. This function sends all packets that contain the associated source or destination address to the Mirror port, regardless of whether the packet is filtered or forwarded.

E **Empty** **2**

When this bit is set and the address is not a multicast address, the address table entry is free to accept a new entry.

DA **Don’t Age** **1**

When this bit is set, the address is never aged. When the Source Port Locking feature has been enabled or aging has been disabled for a given port, the DA bit is considered set for all corresponding address table entries regardless of its actual value.

A **Age** **0**

When The Don't Age (DA) bit is 0, the A bit has the Age function. When functioning as the Age bit, it is cleared to 0 when an address is first learned and each time it is seen as a source address. The A bit is set to1 by the aging logic over the aging interval. If the aging routine discovers the A bit set to 1, the address table entry is cleared by setting the Empty (E) bit to 1. An entry is not aged out when:

1. The DA bit is set to 1 for the address table entry
2. The Port Lock bit is set for the source port in the Source Port Lock register
3. Aging has been disabled for the given port through the Port Aging Control register.

For each of these conditions, the DA bit is considered set and the A bit takes on a new meaning. When Aging has been disabled, the A bit determines if moves are allowed.

Multicast Entries

Multicast Entries use two 64-bit words, which occupy four 32-bit locations.

Lower 64-bit word:

MACA[47:0] **MAC Address** **[59:12]**

This is the 48-bit Multicast address. The address is stored as it is classically written by the user and not as it is transmitted.

VLAN ID[11:0] **VLAN ID** **[11:0]**

This field contains the VLAN ID associated with the IP Multicast MAC address. Specifically, this field is here only to support IP multicast traffic. This arrangement allows any one of the possible VLANs to have a unique port mask for a given multicast address. For non-IP multicast addresses, this field is a don’t care.

Upper 64-bit word:

U **Unused** **[63:47]**
Don't care.

PM[25:0] **Port 1 to Port 26 Mask** **[46:21]**
There is a bit in the port mask for each of the 26 ports. Setting a bit to 1 in the port mask forwards the packet to the corresponding port, assuming all other conditions are satisfied, such as VLAN membership, buffer consumption, and so on.

U **Unused** **[20:13]**
Don't care.

PO **Priority Output (Boost)** **12**
The PO bit determines if the entry corresponds to a multicast group that is to be forwarded at high priority. This only has the effect of placing the packet on the high priority output queue(s). It has no effect on the priority tag field contained in the packet or the Default Port Priority associated with the receiving port. However, this priority bit, when set, overrides the tagged field and the Default Port Priority (PPD) with respect to determining which queue the packet is placed on at each output. When this bit is not set, it has no effect.

MT[4:0] **Multicast Type** **[11:7]**

MT[4:0]	Multicast Type	Description
0b00000	No copy	The packet should not be copied to the CPU
0b00001	Generic Multicast	Copy to CPU
0b00010	IGMP	IGMP packet to be copied to the CPU
0b00011	GVRP/GMRP	VLAN Protocol Packet to be sent to the CPU
0b00100	Unknown Multicast	Unknown Multicast, set by ARL – copied to CPU
0b00101	OSPF	OSPF packet detected by port logic, set by ARL – copied to the CPU

MT[4:0]	Multicast Type	Description
0b00110	PIM	PIM packet detected by port logic, set by ARL, copied to the CPU
0b00111 – 0b11111	Other	Defined by CPU – copied to CPU

Multicast type 4 and 2 will never actually be set in the address table. This entry is set by the ARL

P[1:0] **Priority CPU** **[6:5]**

If the multicast type is other than 0, this field determines on which of the four CPU receive queues the packet will be placed.

P[1:0]	CPU Receive Queue
0b00	Lowest
0b01	Low
0b10	High
0b11	Highest

EMP **Enable Multicast Processing** **4**

When a known multicast packet is received from a port that has a STP state of blocking or learning, and the EMP bit is set in the address table for the multicast entry, the packet is forwarded as if the port were in the forwarding state, except that learning is not performed in the blocking state.

Specifically, if a multicast packet is received from a blocking or learning port, and the EMP bit is in the address table, the packet is forwarded according to the Multicast type and Port Mask. If the EMP bit is set for the Multicast address and the source address entry has the CPU bit set, the packet is forwarded to the CPU.

If the EMP bit is not set or the multicast address is not known, the packet is not sent to the CPU or ports, regardless of the Multicast type field or port mask.

The EMP bit only applies to multicast packets. When a port is in blocking or learning, all unicast and *unknown multicast packets are dropped*, unless the port is a source to a mirror port, in which case, the packet is sent

only to the mirror port. *When in blocking mode, The source address is never learne*d, regardless of the state of the EMP bit.

This function allows the CPU to receive any specified multicast packet on ports that are in blocking or learning STP states.

PM **Port Mirror** **3**

If this bit is set, the packet is also sent to the port specified in the CPU Port Mirror (PM) field of the Port Mirror Mask Register (0x3425.0044), in addition to the intended destination. This function sends all packets that contain the multicast address to the Mirror port, regardless of whether the packet would have been filtered or forwarded.

U **Unused** **[2:0]**

Don't care.

8.13 Transmit Descriptor Engine Registers

The Transmit Descriptor registers are located starting at base address 0x3424.0000.

Register: 0x3424.0000

RX_EN (Receive Queue Enables)
CPU:R/W
Default:0x0000.000F

31 … 4	3	2	1	0
	RX3_EN	RX2_EN	RX1_EN	RX0_EN

RX3_EN **RX Enable for Ring 3** **3**

When this bit is 0, packets are not queued for ring 3. When the bit is 1, packets are queued. The CPU can enable or disable these bits any time.

RX0_EN	Definition
0	Packets are not queued for ring 3.
1	Packets are queued for ring 3 (default)

RX2_EN **RX Enable for Ring 2** **2**

When this bit is 0, packets are not queued for ring 2. When the bit is 1, packets are queued. The CPU can enable or disable these bits any time.

RX0_EN	Definition
0	Packets are not queued for ring 2.
1	Packets are queued for ring 2 (default)

RX1_EN **RX Enable for Ring 1** **1**

When this bit is 0, packets are not queued for ring 1. When the bit is 1, packets are queued. The CPU can enable or disable these bits any time.

RX0_EN	Definition
0	Packets are not queued for ring 1.
1	Packets are queued for ring 1 (default)

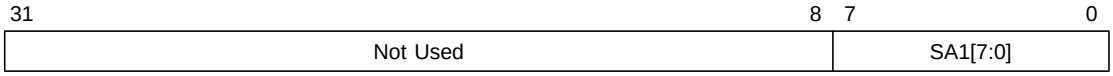
RX0_EN **RX Enable for Ring 0** **0**

When this bit is 0, packets are not queued for ring 0. When the bit is 1, packets are queued. The CPU can enable or disable these bits any time.

RX0_EN	Definition
0	Packets are not queued for ring 0.
1	Packets are queued for ring 0 (default)

Register: 0x3424.0004

RX1_BASE (Base Index for Ring 1)
CPU:R/W
Default:0x0000.0040

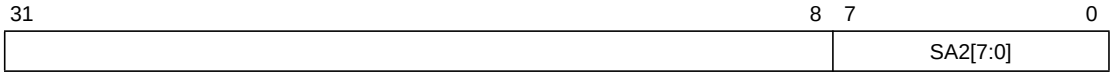


Not Used **[31:8]**
These bits are not used.

SA1[7:0] **Starting Address for Priority 1 Queue** **[7:0]**
Only bits [7:0] are valid. This register contains the starting address of the priority 1 queue (for priority 0 address offset, always starts at 0x00). The default value is 0x40.

Register: 0x3424.0008

RX2_BASE (Base Index for Ring 2)
CPU:R/W
Default:0x0000.0080

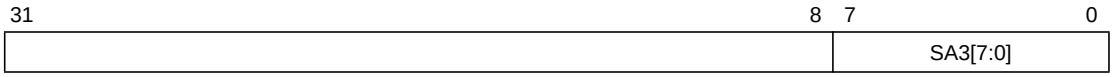


Not Used **[31:8]**
These bits are not used.

SA2[7:0] **Starting Address for Priority 2 Queue** **[7:0]**
This is the same as RX1_BASE, but for the priority 2 queue. Note that SA2[7:0] – 1 serves as the end pointer for the priority 1 queue. The default value is 0x80.

Register: 0x3424.000C

RX3_BASE (Base Index for Ring 3)
CPU:R/W
Default:0x0000.00C0

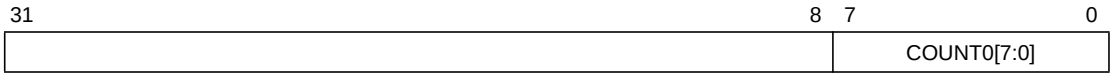


Not Used **[31:8]**
These bits are not used.

SA3[7:0] **Starting Address for Priority 3 Queue** **[7:0]**
This is the same as RX1_BASE, but for the priority 3 queue. Note that SA3[7:0] – 1 serves as the end pointer for the priority 2 queue. The default value is 0xC0.

Register: 0x3424.0010

RX0_FILL
CPU:R/W
Default:0x0000.0001



Not Used **[31:8]**

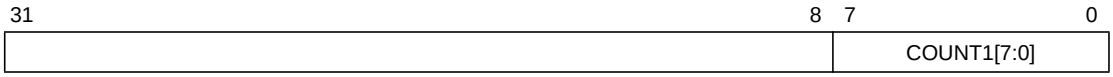
These bits are not used.

COUNT0[7:0] **[7:0]**

Only bits [7:0] are valid. The CPU programs this register at power on and its value should be RX0_BASE[7:0] + 1. Once the register is programmed, the CPU does not update this register. The register is incremented by hardware each time an entry is added to the ring. When this pointer reaches SA1[7:0] – 1 it is set back to 0x00. It stops incrementing when it is equal to NA0[7:0] (FULL condition). The default value is 0x01.

Register: 0x3424.0014

RX1_FILL
CPU:R/W
Default:0x0000.0041



Not Used **[31:8]**

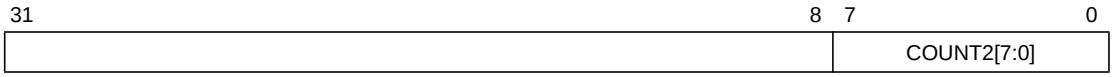
These bits are not used.

COUNT1[7:0] **[7:0]**

Only bits [7:0] are valid. The CPU programs this register at power on and its value should be SA1[7:0] + 1. Once the register is programmed, the CPU does not update this register. The register is incremented by hardware each time an entry is added to the ring. When this pointer reaches SA2[7:0] – 1 it is set back to SA1[7:0]. It stops incrementing when it is equal to NA1[7:0] (FULL condition). The default value is 0x41**.**

Register: 0x3424.0018

RX2_FILL
CPU:R/W
Default:0x0000.0081



Not Used **[31:8]**

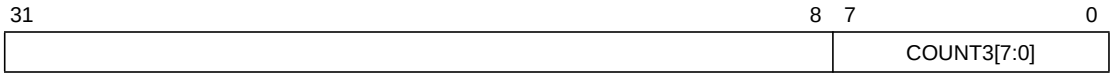
These bits are not used.

COUNT2[7:0] **[7:0]**

Only bits [7:0] are valid. The CPU programs this register at power on and its value should be SA2[7:0] + 1. Once the register is programmed, the CPU does not update this register. The register is incremented by hardware each time an entry is added to the ring. When this pointer reaches SA3[7:0] – 1 it is set back to SA2[7:0]. It stops incrementing when it is equal to NA2[7:0] (FULL condition). The default value is 0x81.

Register: 0x3424.001C

RX3_FILL
CPU:R/W
Default:0x0000.00C1



Not Used **[31:8]**

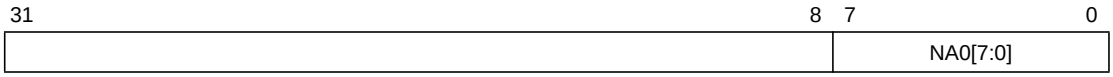
These bits are not used.

COUNT3 **[7:0]**

Only bits [7:0] are valid. The CPU programs this register at power on and its value should be SA3[7:0] + 1. Once the register is programmed, the CPU does not update this register. The register is incremented by hardware each time an entry is added to the ring. When this pointer reaches SA3[7:0] – 1, it is set back to SA2[7:0]. It stops incrementing when it is equal to NA2[7:0] (FULL condition). The default value is 0xC1.

Register: 0x3424.0020

RX0_DRAIN
CPU:R/W
Default:0x0000.0000

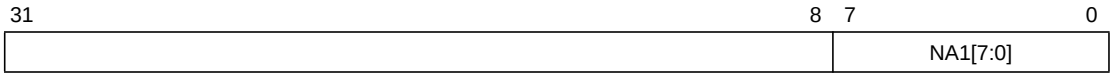


Not Used **[31:8]**
These bits are not used.

NA0[7:0] **Next Available 0** **[7:0]**
Only bits [7:0] are valid. The CPU uses this register to read the next available entry in the ring. The CPU totally maintains this register. It is always initialized to RX0_BASE at power on. The CPU reads entries only when COUNT0[7:0] is at least two entries away from NA0[7:0]. The default value is 0x00.

Register: 0x3424.0024

RX1_DRAIN
CPU:R/W
Default:0x0000.0040

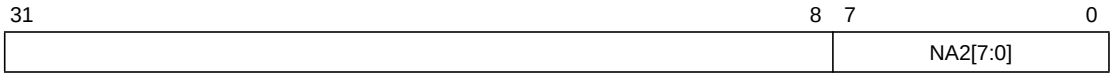


Not Used **[31:8]**
These bits are not used.

NA1[7:0] **Next Available 1** **[7:0]**
Only bits [7:0] are valid. The CPU uses this register to read the next available entry in the ring. The CPU totally maintains this register. It is always initialized to SA1[7:0] at power on. The CPU reads entries only when COUNT1[7:0] is at least two entries away from NA1[7:0]. The default value is 0x40.

Register: 0x3424.0028

RX2_DRAIN
CPU:R/W
Default:0x0000.0080

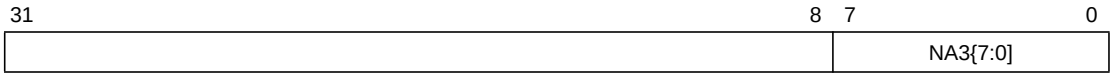


Not Used **[31:8]**
These bits are not used.

NA2[7:0] **Next Available 2** **[7:0]**
Only bits [7:0] are valid. The CPU uses this register to read the next available entry in the ring. The CPU totally maintains this register. It is always initialized to SA2[7:0] at power on. The CPU reads entries only when COUNT2[7:0] is at least two entries away from NA2[7:0]. The default value is 0x80.

Register: 0x3424.002C

RX3_DRAIN
CPU:R/W
Default:0x0000.00C0

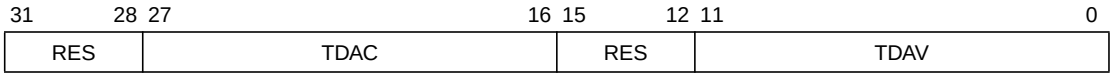


Not Used **[31:8]**
These bits are not used.

NA3[7:0] **Next Available 3** **[7:0]**
Only bits [7:0] are valid. The CPU uses this register to read the next available entry in the ring. The CPU totally maintains this register. It is always initialized to SA3[7:0] at power on. The CPU reads entries only when COUNT3[7:0] is at least two entries away from NA3[7:0]. The default value is 0xC0.

Register: 0x3424.0300

TDAC (Transmit Descriptor Aging Count Register)
CPU:R/W
Default:0x0000.0004



RES **Reserved** **[31:28]**
These bits are reserved.

TDAC[11:0] **Transmit Descriptor Aging Counter** **[27:16]**
TDAC[11:0] is the output of the free running Transmit Descriptor aging counter. This field is used for test purposes only.

RES **Reserved** **[15:12]**
These bits are reserved.

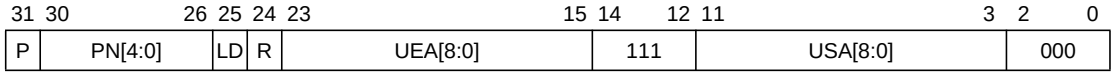
TDAV[11:0] **Transmit Descriptor Aging Value** **[11:0]**
The TDAV[11:0] value is loaded into a down counter that is decremented every 1.0 ms. When the counter reaches 0 it is reloaded with the value in TDAV[11:0] and the aging routine is initiated. A value of 0 disables aging. A value of one results in a 2 ms age interval.

Although the default value is 0x004, it should be set to 0x3E8, which corresponds to 1 second. This avoids unnecessary packet dropping.

Register: 0x3424.0304

DDRPR (Desired Descriptor Ring Pointer Register)
CPU:R/W
Default:0x0000.7000

This register adjusts the start and end pointer for a given Transmit Descriptor ring. A write to this register triggers a request for this adjustment. The Transmit Descriptor Engine (TDE) adjusts when it is convenient to do so. Note that start and end pointers can be changed in the same memory.



P — **Priority Queue to be Loaded** — **31**

P	Description
0	Low Priority
1	High Priority

PN[4:0] — **Port Number** — **[30:26]**
This selects one of the 26 possible ports for which the start and end pointers are to be loaded. Valid values are 1 to 26.

LD — **Load Done** — **25**
This bit is set when the desired start and end pointers are loaded to the desired port and set to the desired priority. This bit gets reset when the CPU writes to this register.

R — **Reserved** — **24**
This bit is reserved.

UEA[8:0] — **Upper 9 Bits of End Address** — **[23:15]**
This field is programmed with the new desired End Pointer value for the descriptor ring indicated by the PN and P fields. This 9-bit value is shifted left 3 bit positions by the ASIC to form a 12-bit pointer. The ASIC then adds 7 to the result to align the pointer to the end of an octal boundary. A 13-bit value is not required since this function can only adjust the size of a ring in its current array. This function should not be used to move a ring to the other array.

111 **These Bits are not Writable** **[14:12]**
They are always 0b111.

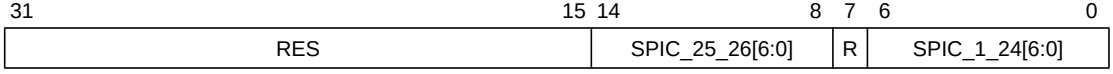
USA[8:0] **Upper 9 Bits of Start Address** **[11:3]**
This field is programmed with the new desired Start Pointer value for the descriptor ring indicated by the PN and P fields. This 9-bit value is shifted left 3 bit positions by the ASIC to form a 12-bit pointer aligned to octal boundaries. A 13-bit value is not required since this function can only adjust the size of a ring in its current array. This function should not be used to move a ring to the other array.

000 **These Bits are not Writable** **[2:0]**
Always 0b000. These bits are treated as the lower three bits of start address.

Register: 0x3424.0308

SPR (Starvation Prevention Register)
CPU:R/W
Default:0x0000.0404

This register determines how many high priority packets are to be transmitted before a low priority packet is transmitted. When this register value is 0, high priority packets are always transmitted and low priority packets are transmitted only if a high priority packet is not available. There are two values, one for ports 1 to 24 and one for ports 25 and 26 (the uplink ports).



RES **Reserved** **[31:15]**
These bits are reserved.

SPIC_25_26[6:0]
Starvation Prevention Interleave Count (25–26) [14:8]
This is the Starvation Prevention Interleave Count for the uplink ports (ports 25 and 26). For every SPIC uplink high priority packet transmitted, one low priority packet is sent. *This value should also apply to trunked ports.* The valid range is 0 to 127. The default value is 0x04.

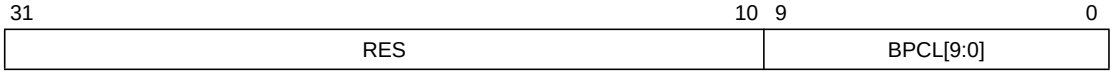
R **Reserved** **7**
This bit is reserved.

SPIC_1_24[6:0]
Starvation Prevention Interleave Count (1–24) [6:0]
This is the Starvation Prevention Interleave Count for ports 1 to 24. For every SPIC high priority packet transmitted, one low priority packet is sent. If a given port happens to be a trunk port, then SPIC_25_26 is used. The valid range is 0 to 127. The default value is 0x04.

Register: 0x3424.0100

BPCL Port 1 (Best Packet Count Limit)
CPU:R/W
Default:0x0000.03FF

This 10-bit register sets the number of broadcast packets allowed during one interval timer. The interval timer is decided by the GBTR (Global Broadcast Timer Register). For details on the GBTR register, see the subsection entitled “GBTR (Global Broadcast Timer Register)” on page 8-326.

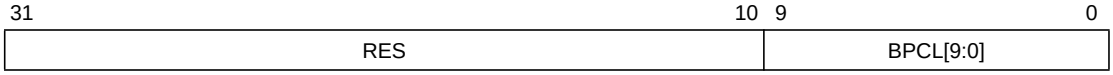


RES **Reserved** **[31:10]**
These bits are reserved.

BPCL[9:0] **Broadcast Packet Count Limit** **[9:0]**
BPCL[9:0] specifies the number of broadcast packets that can be allowed in one interval. If this value is all zeros, all broadcast packets are allowed.

Register: 0x3424.0100 + 4 * (Port Number – 1)

BPCL Port 2–26 (Best Packet Count Limit)
CPU:R/W
Default:0x0000.03FF



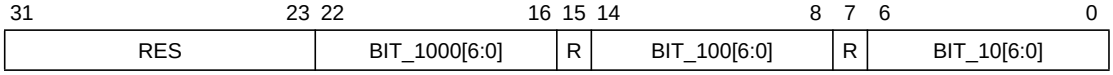
This 10-bit register sets the number of broadcast packets allowed during one interval timer. The interval timer is decided by the GBTR (Global Broadcast Timer Register). For details on the GBTR register, see the subsection entitled “GBTR (Global Broadcast Timer Register)” on page 8-326

RES **Reserved** **[31:10]**
These bits are reserved.

BPCL[9:0] **Broadcast Packet Count Limit** **[9:0]**
BPCL[9:0] specifies the number of broadcast packets that can be allowed in one interval. If this value is all zeros, all broadcast packets are allowed.

Register: 0x3424.0168

GBTR (Global Broadcast Timer Register)
CPU:R/W
Default:0x00xx.xxxx



This register contains three 7-bit fields, one for each type of port (1000 Mbits/s, 100 Mbits/s, and 10 Mbits/s).

RES **Reserved** **[31:23]**
These bits are reserved.

BIT_1000[6:0] **Load Counter 1000** **[22:16]**
This field is used to load the broadcast timer counter for Gigabit ports. The counter is decremented once every 100 µs.

The field is multiplied by 100 µs to form the 1000 Mbits/s broadcast interval. Only the number of broadcast packets specified in the BPCL register are allowed at the given port during this interval.

R **Reserved** **15**
This bit is reserved.

BIT_100[6:0] **Load Counter 100** **[14:8]**
This register is used to load the counter for 100 Mbits/s ports. The counter is decremented once every 1 ms.

This field is multiplied by 1 ms to form the 100 Mbits/s broadcast interval. Only the number of broadcast packets specified in the BPCL register are allowed at the given port during this interval.

R **Reserved** **7**
This bit is reserved.

BIT_10[6:0] **Load Counter 10** **[6:0]**
This register is used to load the counter for 10 Mbits/s ports. This counter is decremented once every 10 ms.

This field is multiplied by 10 ms to form the 10 Mbits/s broadcast interval. Only the number of broadcast packets specified in the BPCL register are allowed at the given port during this interval.

Register: 0x3424.030C

OFCEN (Output Flow Control Enable Register)
CPU:R/W
Default:0x02FF.FFFE

31 27	26	25	24 1	0
RES	OFCEN_26	OFCEN_25	OFCEN_24–1[23:0]	R

Setting these bits will enable output flow control.

RES **Reserved** **[31:27]**
These bits are reserved.

OFCEN_26 **Output Flow Control Enable for Port 26** **26**

OFCEN_26	Description
0	Output flow control disabled (default)
1	Output flow control enabled

OFCEN_25 **Output Flow Control Enable for Port 25** **25**

OFCEN_25	Description
0	Output flow control disabled (default)
1	Output flow control enabled

OFCEN_24–1[23:0]
Output Flow Control Enable for Ports 1 to 24 **[24:1]**

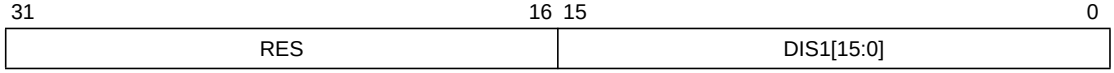
OFCEN_24_1[23:0]	Description
0	Output flow control disabled
1	Output flow control enabled (default)

R **Reserved** **0**
This bit is reserved.

Register: 0x3424.0200

TDAD (Port 1 Transmit Descriptor Aging Discard Count Register)
CPU:R/W
Default:0x0000.0000

This 16-bit counter indicates how many packets have been discarded in port 1 due to aging in the transmit descriptor queue. This includes both the low and high priority queues.



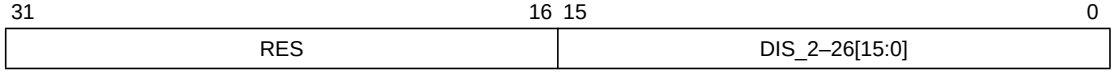
RES **Reserved** **[31:16]**
These bits are reserved.

DIS1[15:0] **Discarded Packet Counter for Port 1** **[15:0]**
This field contains the count of discarded packets for port 1 due to aging.

Register: 0x3424.0200 + 4 * (Port Number – 1)

TDAD (Port 2–26 Transmit Descriptor Aging Discard Count Register)
CPU:R/W
Default:0x0000.0000

This 16-bit counter indicates how many packets have been discarded in ports 2 to 26 due to aging in the transmit descriptor queue. This includes both the low and high priority queue.

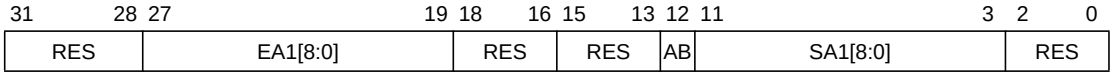


RES **Reserved** **[31:16]**
These bits are reserved.

DIS_2–26[15:0]
Discarded Packet Counter for Ports 2–26 **[15:0]**
This field contains the count of discarded packets for ports 2 to 26 due to aging.

Register: 0x3424.0400

Port 1 Low Priority Start and End Address)
CPU:R/W
Default:0x007F.0000



RES **Reserved** **[31:28]**
These bits are reserved.

EA1[8:0] **Port 1 Low Priority End Address** **[27:19]**
This 9-bit register defines the end location for port 1. The CPU always writes to this register. The lower three bits of the end address are always assumed to be 0b111.

RES **Reserved** **[18:16]**
Always read as 0b111 to indicate the last three bits of the end address.

RES **Reserved** **[15:13]**
Always read as 0b000

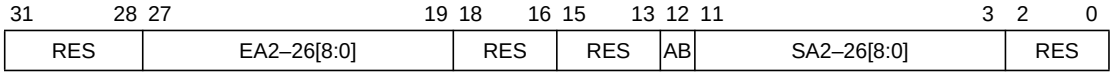
AB **Array Bank** **12**
This bit selects which memory to be used for this port. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

SA1[8:0] **Port 1 Low Priority Start Address** **[11:3]**
This 9-bit register defines starting address for port 1. The CPU always writes to this register. The lower three bits of the start address are always assumed to be 0b000.

RES **Reserved** **[2:0]**
Always read as 0b000.

Register: 0x3424.0500–0x3424.1D00

Port 2–26 Low Priority Start and End Address
CPU:R/W
Default:0x107F.1000, 0x017F.0100, 0x117F.1100, 027F.0200, 0x127F.1200, 0x037F.0300, 0x137F.1300, 0x047F.0400, 0x147F.1400, 0x057F.0500, 0x157F.0500, 0x157F.1500, 0x067F.0600, 0x167F.1600, 0x077F.0700, 0x177F.1700, 0x087F.0800, 0x187F.1800, 0x097F.0900, 0x0A7F.0A00, 0x1A7F.1A00, 0x0B7F.0B00, 0x1B7F.1B00, 0x0CFF.0C00, 0x1CFF.1C00



RES **Reserved** **[31:28]**
These bits are reserved.

EA2–26[8:0] **Port 1 Low Priority End Address** **[27:19]**
This 9-bit register defines the end location for port 2–26. The CPU always writes to this register. The lower three bits of the end address are always assumed to be 0b111.

RES **Reserved** **[18:16]**
Always read as 0b111 to indicate the last three bits of the end address.

RES **Reserved** **[15:13]**
Always read as 0b000

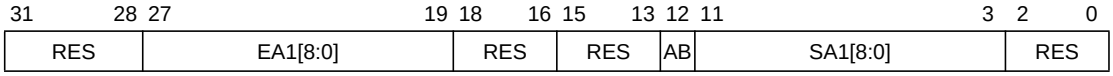
AB **Array Bank** **12**
This bit selects which memory to be used for this port. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

SA2–26[8:0] **Port 1 Low Priority Start Address** **[11:3]**
This 9-bit register defines starting address for port 2–26. The CPU always writes to this register. The lower three bits of the start address are always assumed to be 0b000.

RES **Reserved** **[2:0]**
Always read as 0b000.

Register: 0x3424.040C

Port 1 High Priority Start and End Address
CPU:R/W
Default:0x00FF.0080



RES **Reserved** **[31:28]**
These bits are reserved.

EA1[8:0] **Port 1 High Priority End Address** **[27:19]**
This 9-bit register defines the end location for port 1. The CPU always writes this register. The lower three bits of the end address are always assumed to be 0b111.

RES **Reserved** **[18:16]**
Always read as 0b111 to indicate the last three bits of the end address.

RES **Reserved** **[15:13]**
Always read as 0b000

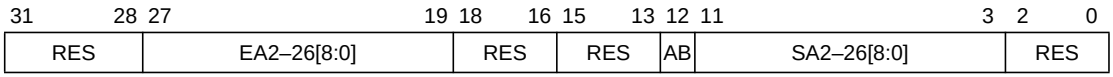
AB **Array Bank** **12**
This bit selects which memory is to be used for this port. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

SA1[8:0] **Port 1 High Priority Start Address** **[11:3]**
This 9-bit register defines the starting address for port 1. The CPU always writes to this register. The lower three bits of the start address are always assumed to be 0b000.

RES **Reserved** **[2:0]**
Always read as 0b000.

Register: 0x3424.050C–0x3424.1D0C

Port 2–26 High Priority Start and End Addresses
CPU:R/W
Default:0x10FF.1000, 0x01FF.0100, 0x11FF.1100, 0x02FF.0200, 0x12FF.1200, 0x03FF.0300, 0x13FF.1300, 0x04FF.0400, 0x14FF.1400, 0x05FF.0500, 0x15FF.1500, 0x06FF.0600, 0x16FF.1600, 0x07FF.0700, 0x17FF.1700, 0x08FF.0800, 0x18FF.1800, 0x09FF.0900, 0x19FF.1900, 0x0aFF.0a00, 0x1AFF.1A00, 0x0BFF.0B00, 0x1BFF.1B00, 0x0DFF.0D00, 0x1DFF.1D00



RES **Reserved** **[31:28]**
These bits are reserved.

EA2–26[8:0] **Port 2–26 High Priority End Address** **[27:19]**
This 9-bit register defines the end location for port 2–26. The CPU always writes this register. The lower three bits of the end address are always assumed to be 0b111.

RES **Reserved** **[18:16]**
Always read as 0b111 to indicate the last three bits of the end address.

RES **Reserved** **[15:13]**
Always read as 0b000

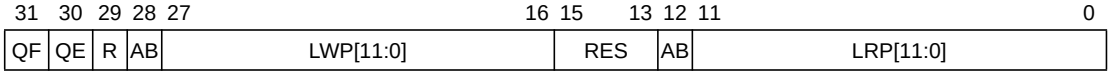
AB **Array Bank** **12**
This bit selects which memory is to be used for each port. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

SA2–26[8:0] **Port 2–26 High Priority Start Address** **[11:3]**
This 9-bit register defines the starting address for port 2–26. The CPU always writes to this register. The lower three bits of the start address are always assumed to be 0b000.

RES **Reserved** **[2:0]**
Always read as 0b000.

Register: 0x3424.0404

Port 1 Low Priority Read and Write Pointer)
CPU:R/W
Default:0x4000.0000



QF **Port 1 Low Priority Queue Full** **31**
QF is the port 1 priority queue full flag.

QE **Port 1 Low Priority Queue Empty** **30**
QE is the port 1 priority queue empty flag.

RES **Reserved** **29**
This bit is reserved.

AB **Array Bank Write** **28**
This bit selects which memory is to be used for this port. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

LWP[11:0] **Port 1 Low Priority Write Pointer** **[27:16]**
LWP[11:0] point to the location where the next descriptor information is written for port 1 priority queue. The CPU can write to this register at any time. This register is incremented every time descriptor information is written for this queue. When this pointer reaches the Port Low End address (EA[8:0]), it is set to the Port Low Start address (SA[8:0]), provided EA[8:0] is not already at SA[8:0]. The LWP[11:0] write pointer increments until it is one entry past the read pointer. At this point, if one more entry is written, the array is treated as full, and comes out of the full condition when the read pointer moves. The PORT_LP_AVAIL intermodule signal (not an I/O signal) is then asserted to inform the switch engine to not generate the forward mask for this queue.

RES **Reserved** **[15:13]**
These bits are reserved.

AB **Array Bank Read** **12**

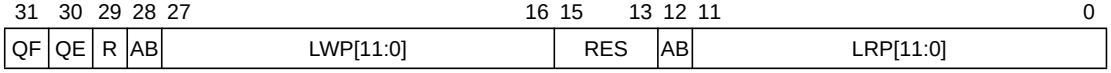
This bit selects which memory is to be used for this port. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

LRP[11:0] **Port 1 Low Priority Read Pointer** **[11:0]**

This 12-bit field points to the location where the next packet descriptor is available to be transmitted from the port queue. Software initializes this field (and LWP[11:0]) to the start pointer. The queue is empty when the read pointer and write pointer are at the same location. When the queue is empty, the PORT_TDE_AVAIL intermodule signal (not an I/O signal) is deasserted to indicate it to the arbiter. This field is incremented every time the port is serviced by driving the packet descriptor from this queue.

Register: 0x3424.0504–0x3424.1D04

Port 2–26 Low Priority Read and Write Pointers
CPU:R/W
Default:0x5000.1000, 0x4100.0100, 0x5100.1100, 0x4200.0200, 0x5200.0200, 0x4300.0300, 0x5300.1300, 0x4400.0400, 0x5400.1400, 0x4500.0500, 0x5500.1500, 0x4600.0600, 0x5600.1600, 0x4700.0700, 0x5700.1700, 0x4800.0800, 0x5800.1800, 0x4900.0900, 0x5900.1900, 0x4A00.0A00, 0x5A00.1A00, 0x4B00.0B00, 0x5B00.1B00, 0x4C00.0C00, 0x5C00.1C00



QF **Port 2–26 Low Priority Queue Full** **31**
QF is the port 2–26 priority queue full flag.

QE **Port 2–26 Low Priority Queue Empty** **30**
QE is the port 2–26 priority queue empty flag.

RES **Reserved** **29**
This bit is reserved.

AB **Array Bank Write** **28**
This bit selects which memory is to be used for each port. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

LWP[11:0] **Port 2–26 Low Priority Write Pointer** **[27:16]**
LWP[11:0] point to the location where the next descriptor information is written for port 2–26 priority queue. The CPU can write to this register at any time. This register is incremented every time descriptor information is written for this queue. When this pointer reaches the Port Low End address (EA[8:0]), it is set to the Port Low Start address (SA[8:0]), provided EA[8:0] is not already at SA[8:0]. The LWP[11:0] write pointer increments until it is one entry past the read pointer. At this point, if one more entry is written, the array is treated as full, and comes out of the full condition when the read pointer moves. The PORT_LP_AVAIL intermodule signal (not an I/O signal) is then asserted to inform the switch engine to not generate the forward mask for this queue.

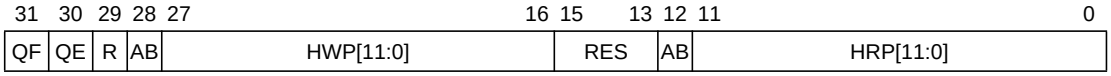
RES **Reserved** **[15:13]**
These bits are reserved.

AB **Array Bank Read** **12**
This bit selects which memory is to be used for each port. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

LRP[11:0] **Port 2–26 Low Priority Read Pointer** **[11:0]**
This 12-bit field points to the location where the next packet descriptor is available to be transmitted from the port queue. Software initializes this field (and LWP[11:0]) to the start pointer. The queue is empty when the read pointer and write pointer are at the same location. When the queue is empty, the PORT_TDE_AVAIL intermodule signal (not an I/O signal) is deasserted to indicate it to the arbiter. This field is incremented every time the port is serviced by driving the packet descriptor from this queue.

Register: 0x3424.0410

Port 1 High Priority Read and Write Pointer
CPU:R/W
Default:0x4080.0080



QF **Port 1 High Queue Full** **31**
QF is the port 1 priority queue full flag.

QE **Port 1 Low Queue Empty** **30**
QE is the port 1 priority queue empty flag.

RES **Reserved** **29**
This bit is reserved.

AB **Array Bank Write** **28**
This bit selects which memory to be used for port 1. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

HWP[11:0] **Port 1 High Priority Write Pointer** **[27:16]**
HWP[11:0] points to the location where the next descriptor information is written for the port 1 priority queue. The CPU can write to this register at any time. This register is incremented every time descriptor information is written for this queue. When this pointer reaches the Port Low End address (EA[8:0]), it is set to the Port Low Start address (SA[8:0]), provided EA[8:0] is not already at SA[8:0]. The HWP[11:0] write pointer increments until it is one entry past the read pointer. At this point, if one more entry is written, the array is treated as full, and comes out of the full condition when the read pointer moves. The PORT_LP_AVAIL intermodule signal (not an I/O signal) is then asserted to inform the switch engine to not generate the forward mask for this queue.

RES **Reserved** **[15:13]**
These bits are reserved.

AB **Array Bank Read** **12**

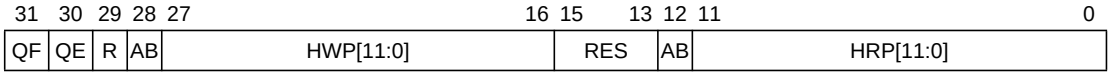
This bit selects which memory is to be used for port 1. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

HRP[11:0] **Port 1 High Priority Read Pointer** **[11:0]**

HRP[11:0] points to the location where the next packet descriptor is available to be transmitted from the port 1 queue. Software initializes this field (and HWP[11:0]) to the start pointer. The queue is empty when the read pointer and write pointer are at the same location. When the queue is empty, the PORT_TDE_AVAIL intermodule signal (not an I/O signal) is deasserted to indicate it to the arbiter. This field is incremented every time the port is serviced by driving the packet descriptor from this queue.

Register: 0x3424.0510–0x3424.1D10

Port 2–26 High Priority Read and Write Pointers
CPU:R/W
Default:0x5080.1080, 0x4180.0180, 0x5180.1180, 0x4280.0280, 0x5280.1280, 0x4380.0380, 0x5380.1380, 0x4480.0480, 0x5480.1480, 0x4580.0580, 0x5580.1580, 0x4680.0680, 0x5680.1680, 0x4780.0780, 0x5780.1780, 0x4880.0880, 0x5880.1880, 0x4980.0980, 0x5980.1980, 0x4A80.0A80, 0x5A80.1A80, 0x4B80.0B80, 0x5B80.1B80, 4C80.0C80, 0x5C80.1C80



QF **Port 2–26 High Queue Full** **31**
QF is the port 2–26 priority queue full flag.

QE **Port 2–26 Low Queue Empty** **30**
QE is the port 2–26 priority queue empty flag.

RES **Reserved** **29**
This bit is reserved.

AB **Array Bank Write** **28**
This bit selects which memory to be used for port 2–26. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

HWP[11:0] **Port 2–26 High Priority Write Pointer** **[27:16]**
HWP[11:0] points to the location where the next descriptor information is written for port 2–26 priority queue. The CPU can write to this register at any time. This register is incremented every time descriptor information is written for this queue. When this pointer reaches the Port Low End address (EA[8:0]), it is set to the Port Low Start address (SA[8:0]), provided EA[8:0] is not already at SA[8:0]. The HWP[11:0] write pointer increments until it is one entry past the read pointer. At this point, if one more entry is written, the array is treated as full, and comes out of the full condition when the read pointer moves. The PORT_LP_AVAIL intermodule signal (not an I/O signal) is then asserted to inform the switch engine to not generate the forward mask for this queue.

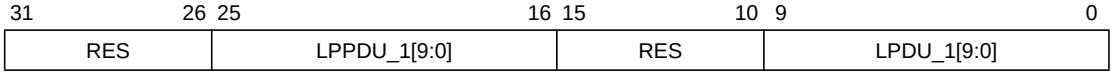
RES **Reserved** **[15:13]**
These bits are reserved.

AB **Array Bank Read** **12**
This bit selects which memory is to be used for port 2–26. If this bit is 0, then memory 1 is used; otherwise memory 2 is used.

HRP[11:0] **Port 2–26 High Priority Read Pointer** **[11:0]**
HRP[11:0] points to the location where the next packet descriptor is available to be transmitted from the port 2–26 queue. Software initializes this field (and HWP[11:0]) to the start pointer. The queue is empty when the read pointer and write pointer are at the same location. When the queue is empty, the PORT_TDE_AVAIL intermodule signal (not an I/O signal) is deasserted to indicate it to the arbiter. This field is incremented every time the port is serviced by driving the packet descriptor from this queue.

Register: 0x3424.0408

Port 1 Low Priority Descriptor Peak and Ring Counter
CPU:R
Default:0x0000.0000



RES **Reserved** **[31:26]**
These bits are reserved.

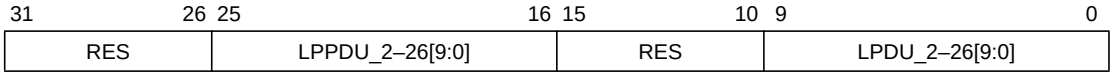
LPPDU_1[9:0]
Low Priority Peak Descriptors Used **[25:16]**
These 10 bits show the peak number of descriptors that were in use since these bits were last read. This field consists of 11 bits (bits [26:16]) for Gigabit ports.

RES **Reserved** **[15:10]**

LPDU_1[9:0] **Low Priority Descriptors Used** **[9:0]**
These 10 bits show the current number of descriptors that are in use in this queue. The CPU reads this field for management purposes. This field consists of 11 bits (bits [10:0]) for Gigabit ports.

Register: 0x3424.0508–0x3424.1D08

Port 2–26 Low Priority Descriptor Peak and Ring Counters
CPU:R
Default:0x0000.0000



RES **Reserved** **[31:26]**
These bits are reserved.

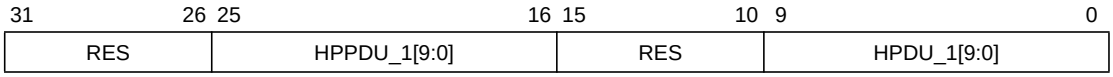
LPPDU_2–26[9:0]
Low Priority Peak Descriptors Used **[25:16]**
These 10 bits show the peak number of descriptors that were in use since these bits were last read. This field consists of 11 bits (bits [26:16]) for Gigabit ports.

RES **Reserved** **[15:10]**

LPDU_2–26[9:0]
Low Priority Descriptors Used **[9:0]**
These 10 bits show the current number of descriptors that are in use in this queue. The CPU reads this field for management purposes. This field consists of 11 bits (bits [10:0]) for Gigabit ports.

Register: 0x3424.0414

Port 1 High Priority Descriptor Peak and Ring Counter
CPU:R
Default:0x0000.0000



RES **Reserved** **[31:26]**
These bits are reserved.

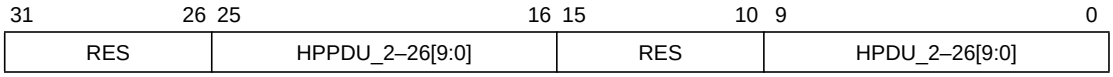
HPPDU_1[9:0]
High Priority Peak Descriptors Used **[25:16]**
These 10 bits show the peak number of descriptors that were in use since these bits were last read. This field consists of 11 bits (bits [26:16]) for Gigabit ports.

RES **Reserved** **[15:10]**
These bits are reserved.

HPDU_1[9:0]
High Priority Descriptors Used **[9:0]**
These 10 bits show the current number of descriptors that are in use in this queue. The CPU reads this field for management purposes. This field consists of 11 bits (bits [10:0]) for Gigabit ports.

Register: 0x3424.0514–0x3424.1D14

Ports 2–26 High Priority Descriptor Peak and Ring Counters
CPU:R
Default:0x0000.0000



RES **Reserved** **[31:26]**
These bits are reserved.

HPPDU_2–26[9:0]
High Priority Peak Descriptors Used **[25:16]**
These 10 bits show the peak number of descriptors that were in use since these bits were last read. This field consists of 11 bits (bits [26:16]) for Gigabit ports.

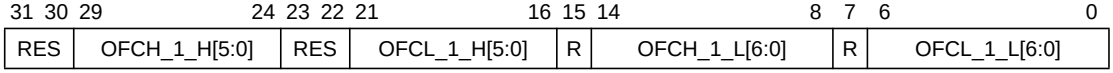
RES **Reserved** **[15:10]**
These bits are reserved.

HPDU_2–26[9:0]
High Priority Descriptors Used **[9:0]**
These 10 bits show the current number of descriptors that are in use in this queue. The CPU reads this field for management purposes. This field consists of 11 bits (bits [10:0]) for Gigabit ports.

Register: 0x3424.0418

OFC1 (Port 1 Output Flow Control Register)
CPU:R/W
Default:0x0000.0000

This register controls output flow based on the port 1 queue. When the number of packets queued for a given ring exceeds the threshold, flow control is applied to the input port that sourced the current packet.



RES **Reserved** **[31:30]**
These bits are reserved.

OFCH_1_H[5:0]
Output Flow Control High Threshold **[29:24]**
This is the high threshold for the high priority ring of port 1. The content of this field is multiplied by 8 to form the output flow control high threshold for the port 1 high priority ring. When HPPDU_1[9:0] ≥ OFCH_1_H and the OFCEN bit (output flow control enable) is set for the port, flow control is initiated at any input port that attempts to add a packet to the output queue.

For uplink ports, bits [30:24] are used and bit [31] is reserved.

RES **Reserved** **[23:22]**
These bits are reserved.

OFCL_1_H[5:0]
Output Flow Control Low Threshold **[21:16]**
This is the low threshold for the high priority ring of port 1. The content of this field is multiplied by 8 to form the output flow control low threshold for the port 1 high priority ring. When HPPDU_1[9:0] < OFCL_1_H and the OFCEN bit (output flow control enable) is set for the port, flow control is terminated at all affected input ports.

For uplink ports, bits [22:16] are used and bit [23] is reserved.

RES **Reserved** **15**
This bit is reserved.

OFCH_1_L[6:0]

Output Flow Control High Threshold [14:8]

This is the high threshold for the low priority ring of port 1. The content of this field is multiplied by 8 to form the output flow control high threshold for the port 1 low priority ring. When LPPDU_1[9:0] ≥ OFCH_1_L and the OFCEN bit (output flow control enable) is set for the port, flow control is initiated at any input port that attempts to add a packet to the output queue.

For uplink ports, bits [15:8] are used.

R **Reserved** **7**

This bit is reserved.

OFCL_1_L[6:0]

Output Flow Control Low Threshold [6:0]

This is the low threshold for the low priority ring of port 1. The content of this field is multiplied by 8 to form the output flow control low threshold for the port 1 low priority ring. When LPPDU_1[9:0] < OFCH_1_L and the OFCEN bit (output flow control enable) is set for the port, flow control is terminated at all affected input ports.

For uplink ports, bits [7:0] are used.

Register: 0x3424.0518–0x3424.1D18

OFC 2–26 (Port 2–26 Output Flow Control Registers)
CPU:R/W
Default:0x0000.0000

31 30	29 24	23 22	21 16	15	14 8	7	6 0
RES	OFCH_2–26_H[5:0]	RES	OFCL_2–26_H[5:0]	R	OFCH_2–26_L[6:0]	R	OFCL_2–26_L[6:0]

RES **Reserved** **[31:30]**
These bits are reserved.

OFCH_2–26_H[5:0]
Output Flow Control High Threshold **[29:24]**
This is the high threshold for the high priority ring of port 2–26. The content of this field is multiplied by 8 to form the output flow control high threshold for the port 1 high priority ring. When HPPDU_2–26[9:0] ≥ OFCH_2–26_H and the OFCEN bit (output flow control enable) is set for the port, flow control is initiated at any input port that attempts to add a packet to the output queue.

For uplink ports, bits [30:24] are used and bit [31] is reserved.

RES **Reserved** **[23:22]**
These bits are reserved.

OFCL_2–26_H[5:0]
Output Flow Control Low Threshold **[21:16]**
This is the low threshold for the high priority ring of port 2–26. The content of this field is multiplied by 8 to form the output flow control low threshold for the port 2–26 high priority ring. When HPPDU_2–26[9:0] < OFCL_2–26_H and the OFCEN bit (output flow control enable) is set for the port, flow control is terminated at all affected input ports.

For uplink ports, bits [22:16] are used and bit [23] is reserved.

RES **Reserved** **15**
This bit is reserved.

OFCH_2–26_L[6:0]

Output Flow Control High Threshold **[14:8]**

This is the high threshold for the low priority ring of port 2–26. The content of this field is multiplied by 8 to form the output flow control high threshold for the port 2–26 low priority ring. When LPPDU_2–26[9:0] ≥ OFCH_2–26_L and the OFCEN bit (output flow control enable) is set for the port, flow control is initiated at any input port that attempts to add a packet to the output queue.

For uplink ports, bits [15:8] are used.

R **Reserved** **7**

This bit is reserved.

OFCL_2–26_L[6:0]

Output Flow Control Low Threshold **[6:0]**

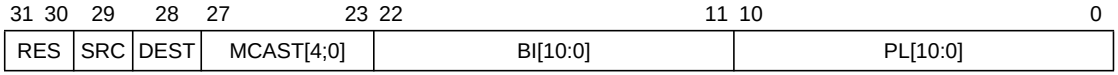
This is the low threshold for the low priority ring of port 2–26. The content of this field is multiplied by 8 to form the output flow control low threshold for the port 2–26 low priority ring. When LPPDU_2–26[9:0] < OFCH_2–26_L and the OFCEN bit (output flow control enable) is set for the port, flow control is terminated at all affected input ports.

For uplink ports, bits [7:0] are used.

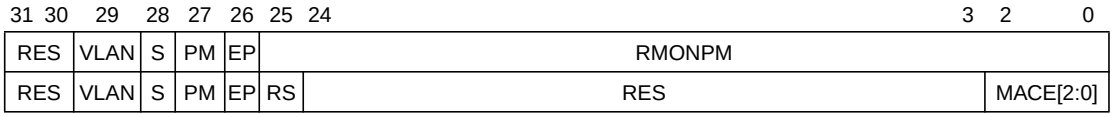
Register: 0x3424.4000

Receive Descriptor Memory
CPU:R/W
Default:0x0000.0000

First Word



Second Word



Normally, hardware writes receive descriptors to this memory at address RXn_FILL[7:0]. The CPU then reads the memory at address RXn_DRAIN[7:0]. Because each entry is 64 bits wide, the CPU reads the descriptor with two 32-bit accesses. During the test mode, the CPU also writes to these locations.

The actual address for a given receive descriptor entry is as follows:

First Word Address:

tde_base + 4000 + NAX[7:0] * 8 + 0x0

Second Word Address:

tde_base + 4000 + NAX[7:0] * 8 + 0x4

First Word

RES **Reserved** **[31:30]**
These bits are reserved.

SRC **CPU SRC** **29**
This bit indicates the CPU bit was set in a unicast address entry found during the source address lookup.

DEST **CPU DEST** **28**
This bit indicates the CPU bit was set in a unicast address entry found during the destination address lookup.

MCAST[4:0] **Multicast Type** **[27:23]**
If the packet is multicast or broadcast, this field contains one of the nonzero multicast type values from the Multicast address table entry. This field is 0 for unicast packets. Software can reserve a special multicast type code for the Broadcast destination address.

BI[10:0] **Buffer Index** **[22:11]**
This is the packet buffer index. The address of the buffer can be calculated by multiplying the buffer index by 1536, then adding the based address of the packet buffer space. Up to 4096 packet buffers are supported.

PL[10:0] **Packet Length** **[10:0]**
The length of the packet in bytes. Bytes are counted from the first byte of the destination address through the last byte of the Frame Check Sequence (FCS). It does not include the four bytes of data (source port) before the packet.

Second Word

RES **Reserved** **[31:30]**
These bits are reserved.

VLAN **CPU VLAN** **29**
This bit indicates the CPU has set the CPU bit in the VLAN table and the packet is being received due to VLAN monitoring to the CPU.

S **Security** **28**
When set, this bit indicates the packet is the result of a source port mismatch security violation. The source port was configured for security and a packet with an unknown source address was received or a packet with a source address not previously known to reside on this port was received.

PM **CPU Port Mirror** **27**

When set, this bit indicates the CPU has configured itself as the mirror port and the packet is being received due to port or address mirroring.

EP **Error Packet** **26**

EP	Description
0	Error
1	No Error

If the EP bit is 0:

RMONPM[25:0]

RMON Port Mask **[25:0]**

Bits set in this mask indicate that the extended RMON sample countdown register(s) transitioned to zero on the specified port(s), and the packet is to be considered as a sample for the specified port(s). Bits can be set for any of the output ports, as well as for the input port. As with all other types of packet receptions, the physical input port number is stored in the first byte of the buffer.

If the EP bit is 1:

RS **Input Port RMON Sample** **25**

When set, this bit indicates the packet extended RMON sample countdown register transitioned to zero on the input port (the input port is indicated by the source port number in the first byte of the packet buffer), and the packet is to be considered a sample.

RES **Reserved** **[24:3]**

These bits are reserved.

MACE[2:0] **Mac Error Bits** **[2:0]**

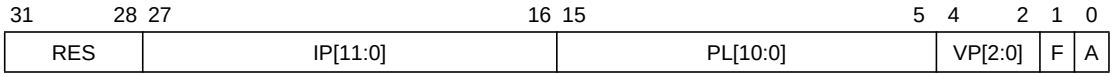
For packets with errors (EP = 1), the MACE[2:0] field indicates the cause of the error. If this field is 0, a CRC error has occurred on a packet 64–1522 (1518 if untagged) bytes long.

Bit 0: Frame alignment error (and CRC was bad).
Bit 1: Symbol error (not detectable in 10 Mbits/s mode)
Bit 2: Oversize (packet was larger than 1522 (1518 if untagged) bytes. CRC/alignment status is unknown.
Bits [24:3]: The ASIC always writes 0 to these bits.

Note: Frame alignment errors cannot occur on Gigabit links.

Register: 0x3424.8000–0x3424.B7FC

Descriptor Memory Group 1
CPU:R/W
Default:0xxxxx.xxxx



The registers make up the descriptor memory for group 1. This memory stores descriptor information for Port 1 through Port 12, and Port 25. The total memory from the CPU point of view is 16 Kbytes. The CPU can read and write these locations as 32-bit accesses only.

Note: The CPU should never directly access the descriptor memory except for memory testing purposes.

RES **Reserved** **[31:28]**
These bits are reserved.

IP[11:0] **Index Pointer** **[27:16]**
This contains the pointer to the buffer that contains the packet.

PL[10:0] **Packet Length** **[15:5]**
This field contains the number of bytes from the first byte of the Destination address to the last byte of the FCS. The port logic uses the field to determine when the last byte of a packet has been read from memory.

VP[2:0] **Virtual Packet** **[4:2]**
The port logic uses the VP[2:0] processing field to determine if a VLAN and/or priority tag should be added, removed, modified, or passed unaltered. The VP field is generated from the extracted packet data and the VLAN table. The VLAN transmit port operation is summarized in the table.

VP[2:0]	Operation	CRC Recalculation
0b000	Pass Unaltered	No
0b001	Add PVID + PPD	Yes
0b010	Remove Tag	Yes
0b011	Recalculate CRC	Yes

VP[2:0]	Operation	CRC Recalculation
0b100	Reserved	Yes
0b101	Modify with PVID	Yes
0b110	Modify with PPD	Yes
0b111	Modify with PVID + PPD	Yes

F **Flush**

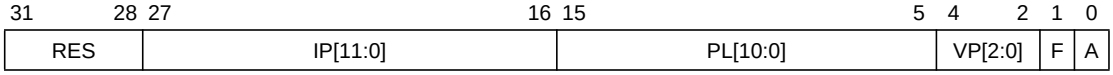
The Transmit descriptor aging logic (TDAL) sets this bit when an entry has been in the transmit descriptor ring longer than 1 second. When descriptor entries are initially created, both the Flush and the Age bits are reset. When the port logic pulls a descriptor with the Flush bit set, the port logic returns the associated index to the ABIT and does not transmit the packet. The next descriptor in the ring is then requested.

A **Age** **0**

The Transmit Descriptor Aging Logic (TDAL) uses this bit. New descriptor entries have both the Flush and Age bit reset. The TDAL parses the table every second. The Logic sets Age bits that are 0 to 1. The Flush bit is set for entries found that are already set. When the port logic pulls in a descriptor from the table that has the Flush bit set, the port returns the index back to the ABIT and does not transmit the associated packet. The standard specifies a maximum transit delay of 4 seconds with a recommended time of 1 second. This mechanism ages entries on the order of 1 to 2 seconds. The Age bit is only used by the descriptor logic and is not passed to the port logic. Note that the TDAL is the absolute lowest priority access into the table. The TDAL parses each ring from the read pointer to the write pointer every second.

Register: 0x3424.C000–0x3424.F7FC

Descriptor Memory Group 2
CPU:R/W
Default:0xxxxx.xxxx



The registers make up the descriptor memory for group 2. This memory stores descriptor information for Port 2 through Port 24, and Port 26.The total memory from the CPU point of view is 16 Kbytes. The CPU can read and write these locations as 32-bit accesses only.

Note: The CPU should never directly access the descriptor memory except for memory testing purposes.

RES **Reserved** **[31:28]**
These bits are reserved.

IP[11:0] **Index Pointer** **[27:16]**
This contains the pointer to the buffer that contains the packet.

PL[10:0] **Packet Length** **[15:5]**
This field contains the number of bytes from the first byte of the Destination address to the last byte of the FCS. The port logic uses the field to determine when the last byte of a packet has been read from memory.

VP[2:0] **Virtual Packet** **[4:2]**
The port logic uses the VP[2:0] processing field to determine if a VLAN and/or priority tag should be added, removed, modified, or passed unaltered. The VP field is generated from the extracted packet data and the VLAN table. The VLAN transmit port operation is summarized in the table.

VP[2:0]	Operation	CRC Recalculation
0b000	Pass Unaltered	No
0b001	Add PVID + PPD	Yes
0b010	Remove Tag	Yes
0b011	Recalculate CRC	Yes

VP[2:0]	Operation	CRC Recalculation
0b100	Reserved	Yes
0b101	Modify with PVID	Yes
0b110	Modify with PPD	Yes
0b111	Modify with PVID + PPD	Yes

F **Flush** **1**

The Transmit descriptor aging logic (TDAL) sets this bit when an entry has been in the transmit descriptor ring longer than 1 second. When descriptor entries are initially created, both the Flush and the Age bits are reset. When the port logic pulls a descriptor with the Flush bit set, the port logic returns the associated index to the ABIT and does not transmit the packet. The next descriptor in the ring is then requested.

A **Age** **0**

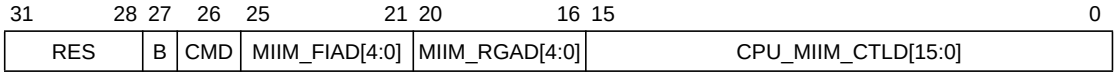
The Transmit Descriptor Aging Logic (TDAL) uses this bit. New descriptor entries have both the Flush and Age bit reset. The TDAL parses the table every second. The Logic sets Age bits that are 0 to 1. The Flush bit is set for entries found that are already set. When the port logic pulls in a descriptor from the table that has the Flush bit set, the port returns the index back to the ABIT and does not transmit the associated packet. The standard specifies a maximum transit delay of 4 seconds with a recommended time of 1 second. This mechanism ages entries on the order of 1 to 2 seconds. The Age bit is only used by the descriptor logic and is not passed to the port logic. Note that the TDAL is the absolute lowest priority access into the table. The TDAL parses each ring from the read pointer to the write pointer every second.

8.14 MII Management (MIIM) Control Register

This section describes the MIIM Control register.

Register: 0x3421.0000

MIIM_CTL1
CPU:R/W
Default:0x0000.0000



RES **Reserved** **[31:28]**
These bits are reserved.

B **CPU MIIM Transfer is Busy (Read Only)** **27**

B	Description
0	The previous CPU MIIM transfer has fully completed and the MIIM is ready for another operation.
1	The CPU MIIM transaction has not completed

CMD **CPU MIIM Command** **26**

CMD	Description
0	CPU is initiating a read
1	CPU is initiating a write

MIIM_FIAD[4:0]
CPU MIIM PHY Address **[25:21]**
This field specifies the address of the PHY the CPU has selected for reading or writing (port 1 uses PHY address 1).

MIIM_RGAD[4:0]
CPU MIIM PHY Register Address **[20:16]**
This field designates the address of the register being read or written within the PHY.

CPU_MIIM_CTLD[15:0]
MIIM Control Registers **[15:0]**
For reads from the PHY, this field contains the data read from the PHY through the transceiver. For writes to the PHY, the CPU places data in this field for the next write to the PHY through the transceiver.

Chapter 9
Specifications

This chapter provides the specifications of the L64324 24 x 2 Fast Ethernet Intelligent Switch and contains the following sections:

9.1 Absolute Maximum Ratings

Table 9.1 shows the absolute maximum ratings. These limits, if exceeded, could cause permanent damage to the device or affect device reliability. All voltages are with respect to ground unless otherwise specified.

Table 9.1 Absolute Maximum Ratings[1]

Symbol	Parameter	Limits[1]	Unit
V_{DD}	DC Core Supply Voltage[2]	–0.3 to +3.1	V
$V_{DD3.3}$	DC I/O Supply Voltage[3]	–0.3 to +3.9	V
V_{IN}	3.3 V Drive Input Voltage[3]	–1.0 to $V_{DD3.3}$ +0.0	V
I_{IN}	DC Input Current	±10	mA
T_{STG}	Storage Temperature Range (Plastic)[4]	–65 to +125	°C

1. Ratings in this table are those beyond which permanent device damage is likely to occur. These values should not be used as limits for normal device operation. Reference for these values is V_{SS}.
2. Internal cells (core) operate at 2.5 V.
3. I/O cells operate at 3.3 V.
4. Package is E-PBGA 388.

9.2 Recommended Operating Conditions

Table 9.2 shows the recommended operating conditions for the device.

Table 9.2 Recommended Operating Conditions

Symbol	Parameter	Limits[1]	Unit
V_{DD}	DC Core Supply Voltage[2]	+2.375 to +2.625	V
$V_{DD3.3}$	DC I/O Supply Voltage[3]	+3.0 to +3.6	V
T_J	Operating Junction Temperature Range	–40 to +125	°C

1. For normal device operation adhere to the limits in this table. Sustained operation of a device at conditions exceeding these values, even if they are within the absolute maximum rating limits, may result in permanent device damage or impaired device reliability. Device functionality to stated DC and AC limits is not guaranteed if conditions exceed recommended operating conditions.
2. Core Supply voltage is 2.5 V nominal.
3. I/O Supply voltage is 3.3 V nominal.

9.3 DC Characteristics

Table 9.3 lists the device DC electrical characteristics..

Table 9.3 DC Characteristics

Symbol	Parameter	Condition[1]	Minimum	Typical	Maximum	Unit
$V_{DD3.3}$	DC I/O Supply Voltage[2]	–	3.135	3.3	3.465	V
V_{IL}	Input Low Voltage	–	$V_{DD3.3}$ – 0.5	–	0.8	V
V_{IH}	Input High Voltage	–	2.0	–	$V_{DD3.3}$ + 0.3	V
V_T	Switching Threshold	–	-	1.4	2.0	V
V_{T+}	Schmitt Trigger, Positive going Threshold	–	-	1.7	2.0	V
V_{T-}	Schmitt Trigger, Negative going Threshold	–	0.8	1.0	–	V
	Schmitt Trigger, Hysteresis	–	0.6	0.7	–	V
I_{IN}	Input Current	$V_{IN} = V_{DD3.3}$ or V_{SS}	–10	1	10	µA
	Inputs with Pull-down Resistors	$V_{IN} = V_{DD3.3}$	35	115	222	µA
	Inputs with Pull-up Resistors	$V_{IN} = V_{SS}$	–35	–115	–214	µA
V_{OH}	Output High Voltage	Commercial		–		
	Type b1	I_{OH} = -1mA	2.4		$V_{DD3.3}$	V
	Type b2	I_{OH} = -2mA	2.4		$V_{DD3.3}$	V
	Type bz25	I_{OH} = TBD mA	2.4		$V_{DD3.3}$	V
	Type bz50	I_{OH} = TBD mA	2.4		$V_{DD3.3}$	V
	Type bz75	I_{OH} = TBD mA	2.4		$V_{DD3.3}$	V

Table 9.3 DC Characteristics (Cont.)

Symbol	Parameter	Condition[1]	Minimum	Typical	Maximum	Unit
V_{OL}	Output Low Voltage	Commercial		–		
	Type b1	I_{OH} = 1mA		0.2	0.4	V
	Type b2	I_{OH} = 2mA		0.2	0.4	V
	Type bz25	I_{OH} = TBD mA		0.2	0.4	V
	Type bz50	I_{OH} = TBD mA		0.2	0.4	V
	Type bz75	I_{OH} = TBD mA		0.2	0.4	V
I_{OZ}	3-state Output Leakage Current	$V_{OH} = V_{SS}$ or $V_{DD3.3}$	–10	1	10	µA
I_{OS}	Output Short Circuit Current[3], bz50	$V_O = V_{DD3.3}$ $V_O = V_{SS}$	–	–	TBD	mA
	Output Short Circuit bz50	$V_O = V_{DD3.3}$ $V_O = V_{SS}$	–	–	TBD	mA
C_{IN}	Input Capacitance[3]	Input and Bidirectional Buffers		4.0		pF
C_{OUT}	Output Capacitance[3]	Output buffer[3]		4.0		pF
W_D	Power Dissipation	V_{DD},$V_{DD3.3}$ = max; Sys_clk, Mac_clk = 125 MHz			6.0	W

1. Junction Temperature Range –40 to +125 °C, ±5% Power Supply.
2. Output using single buffer structure (excluding package).
3. Excluding package capacitance.

9.4 AC Characteristics

The device contains eight groups of pins for which the AC timing is specified. These eight groups are:

- Clocks and Reset
- Serial Media Independent Interface (SMII)
- Gigabit Media Independent Interface (GMII)

- Ten Bit Interface (TBI)
- Media Independent Interface (MII)
- Management Interface
- SDRAM Interface
- LED Interface

9.4.1 Clocks and Reset

The L64324 has two primary clock inputs: SYS_CLK and MAC_CLK. SYS_CLK is the main system clock, which operates between 125 MHz and 143 MHz. MAC_CLK is used by the E110 MAC and GMAC port logic to interface to the PHY devices. This clock always runs at 125 MHz. Each of the GMAC ports also have one receive clock input. The uplink clocks (RXCLK_GRXCLK25 and RXCLK_GRXCLK26) operate at 125 MHz in GMII mode. The device has one active-LOW reset input and one active-LOW reset output. The reset input RESET_INn is the main system reset. The output RESET_OUT can become active due to the internal watchdog reset going active. Figure 9.1 shows the timing diagram and Table 9.4 lists the parameters of the clocks and reset signals.

Figure 9.1 Clock and Reset Timing Diagram

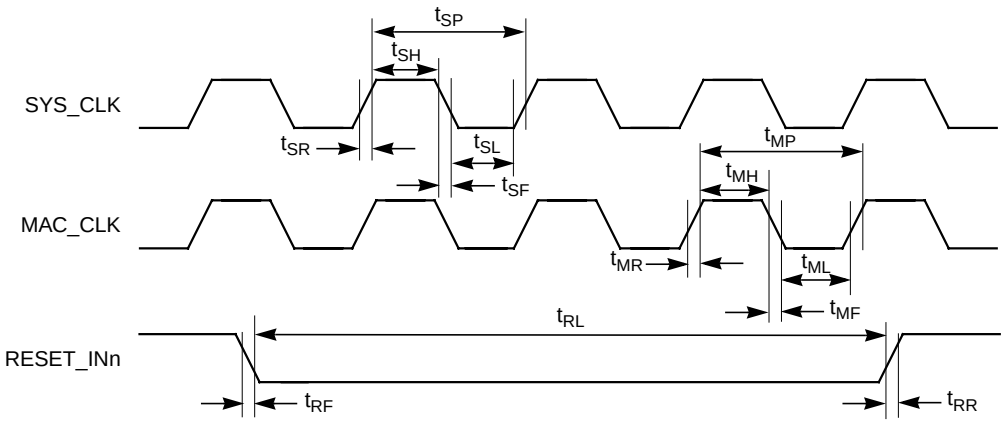


Table 9.4 Clock/Reset Timing

Symbol	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{SR}	SYS_CLK rise time	125 MHz operation			1	ns	
t_{SF}	SYS_CLK fall time	125 MHz operation			1	ns	
t_{SH}	SYS_CLK HIGH time	125 MHz operation	3.5		4.5	ns	
t_{SL}	SYS_CLK LOW time	125 MHz operation	3.5		4.5	ns	
t_{SP}	SYS_CLK period	125 MHz operation		8.0		ns	± 100 PPM
t_{MR}	MAC_CLK rise time	125 MHz			1	ns	
t_{MF}	MAC_CLK fall time	125 MHz			1	ns	
t_{MH}	MAC_CLK HIGH time	125 MHz	3.5		4.5	ns	
t_{ML}	MAC_CLK LOW time	125 MHz	3.5		4.5	ns	
t_{MP}	MAC_CLK period	125 MHz		8.0		ns	± 100 PPM
t_{RF}	RESET_INn fall time				1	ns	
t_{RR}	RESET_INn rise time				1	ns	
t_{RL}	RESET_INn LOW (active) time		1			ms	

9.4.2 Serial Media Independent Interface (SMII)

The L64324 interface to the PHY 10/100 devices is the SMII interface. The uplink ports also have a 10/100 MAC and can be configured to operate in SMII mode. The SMII has one output and one input per port and two reference outputs for each six-port group. These reference outputs are the SYNC and CLK signals. Each port has a TXD output and RXD input. The timing reference is the CLK output. Figure 9.2 shows the timing diagram and Table 9.5 lists the parameters.of the Serial Media Independent Interface (SMII).

Figure 9.2 SMII Timing Diagram

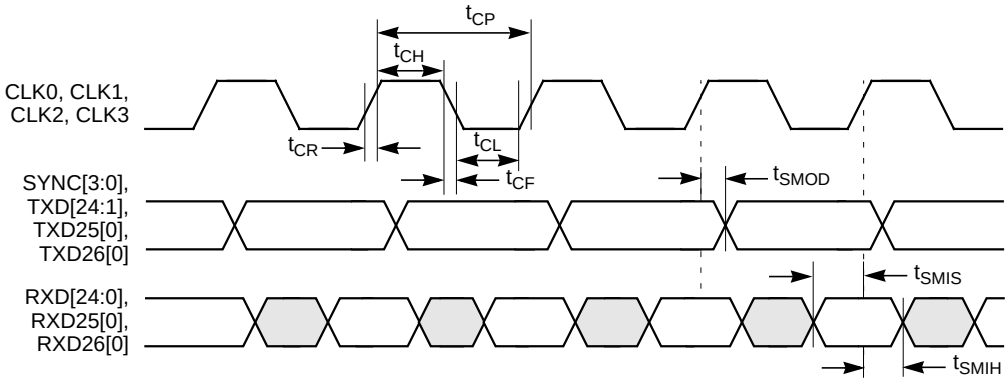


Table 9.5 SMII Timing

Symbol	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{CR}	SMII clock rise time				1	ns	
t_{CF}	SMII clock fall time				1	ns	
t_{CH}	SMII clock HIGH time		3.5		4.5	ns	
t_{CL}	SMII clock LOW time		3.5		4.5	ns	
t_{CP}	SMII clock period			8.0		ns	± 100 PPM
t_{SMOD}	SMII Output Delay	Positive edge of SMII clock	2.0		6.0	ns	
t_{SMIS}	SMII Input Setup	Positive edge of SMII clock	1.0			ns	
t_{SMIH}	SMII Input Hold	Positive edge of SMII clock	0.5			ns	

9.4.3 Gigabit Media Independent Interface (GMII)

The L64324 has two GMAC uplink ports; port 25 and port 26. These uplink ports can operate in 1000BASE-X mode or 10/100BASE-T mode. When the uplink ports are operated in 1000BASE-X mode, the GMII or TBI interface is used. When the ports operate in 10/100 BASE-T mode, the MII interface or the SMII interface is used. In all these modes, the

same pins of the ports are multiplexed to perform different functions. The timing specified is applicable to pins depending on the mode of operation. Figure 9.3 shows the timing diagram and Table 9.6 lists the parameters of the Gigabit Media Independent Interface (GMII)

Figure 9.3 GMII Timing Diagram

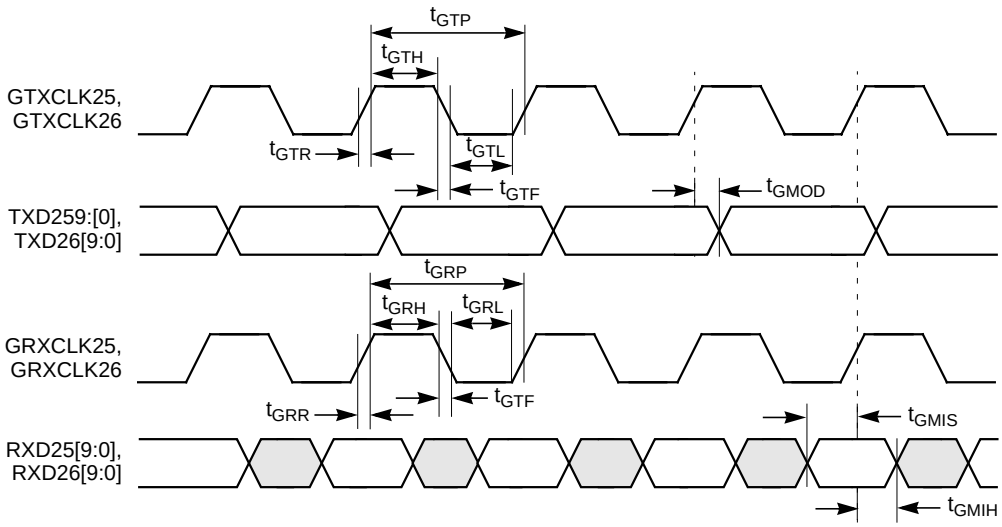


Table 9.6 GMII Timing

Symbol	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{GTR}	GMII Tx clock rise time				1	ns	
t_{GTF}	GMII Tx clock fall time				1	ns	
t_{GTH}	GMII Tx clock HIGH time		2.5			ns	
t_{GTL}	GMII Tx clock LOW time		2.5			ns	
t_{GTP}	GMII Tx clock period			8.0		ns	± 100 PPM
t_{GRR}	GMII Rx clock rise time				1	ns	
t_{GRF}	GMII Rx clock fall time				1	ns	

Table 9.6 GMII Timing (Cont.)

Symbol	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{GRH}	GMII Rx clock HIGH time		2.5			ns	
t_{GRL}	GMII Rx clock LOW time		2.5			ns	
t_{GRP}	GMII Rx clock period			8.0		ns	± 100 PPM
t_{GMOD}	GMII Tx Output Delay	Positive edge of GMII Tx clock	1.0		5.5	ns	
t_{GMIS}	GMII Rx Input Setup	Positive edge of GMII Tx clock	2.5			ns	
t_{GMIH}	GMII Rx Input Hold	Positive edge of GMII Tx clock	0.5			ns	

9.4.4 Ten Bit Interface (TBI)

Ten bit interface (TBI) timing applies to the pins of the GMAC uplink ports (port 25 and port 26) when they are used in 1000BASE-X mode and TBI is selected instead of GMII. Figure 9.4 shows the timing diagram and Table 9.7 lists the TBI timing parameters.

Figure 9.4 TBI Timing Diagram

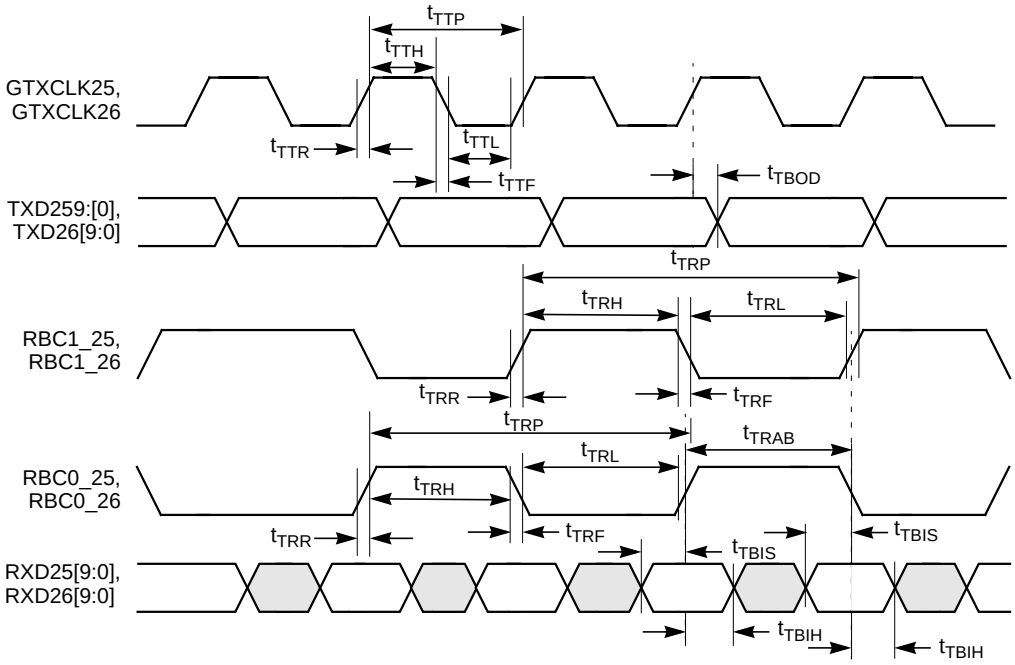


Table 9.7 TBI Timing

Sym	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{TTR}	TBI Tx clock rise time				1	ns	
t_{TTF}	TBI Tx clock fall time				1	ns	
t_{TTH}	TBI Tx clock HIGH time		3.2			ns	
t_{TTL}	TBI Tx clock LOW time		3.2			ns	
t_{TTP}	TBI Tx clock period			8.0		ns	± 100 PPM
t_{TRR}	TBI Rx clock rise time				1	ns	
t_{TRF}	TBI Rx clock fall time				1	ns	
t_{TRH}	TBI Rx clock HIGH time		2.5			ns	

Table 9.7 TBI Timing (Cont.)

Sym	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{TRL}	TBI Rx clock LOW time		2.5			ns	
t_{TRP}	TBI Rx clock period			16.0		ns	
t_{TRAB}	TBI RX clock skew		7.5		8.5	ns	
t_{TBOD}	TBI Tx Output Delay	Positive edge of GMII Tx clock	1.0		5.5	ns	
t_{TBIS}	TBI Rx Input Setup	Positive edge of GMII Tx clock	2.5			ns	
t_{TBIH}	TBI Rx Input Hold	Positive edge of GMII Tx clock	1.5			ns	

9.4.5 Media Independent Interface (MII)

The pins of the GMAC uplink ports can be made to operate in MII mode when the uplink port is operated in 10/100 mode. When operating in MII mode the timing specified in Figure 9.5 and Table 9.8 are applicable to respective pins.

Figure 9.5 MII Timing Diagram

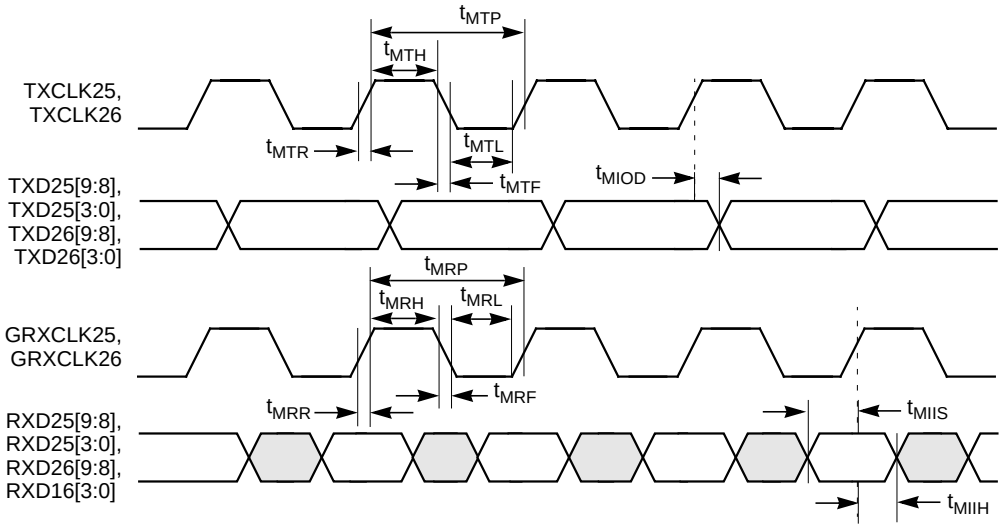


Table 9.8 MII Timing

Sym	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{MTR}	MII Tx clock rise time			2		ns	
t_{MTF}	MII Tx clock fall time			2		ns	
t_{MTH}	MII Tx clock HIGH time	100BASE-X	14			ns	
t_{MTL}	MII Tx clock LOW time	100BASE-X	14			ns	
t_{MTH}	MII Tx clock HIGH time	10BASE-X	140			ns	
t_{MTL}	MII Tx clock LOW time	10BASE-X	140			ns	
t_{MTP}	MII Tx clock period	100BASE-X		40.0		ns	± 100 PPM
t_{MTP}	MII Tx clock period	10BASE-X		400.0		ns	± 100 PPM
t_{MRR}	MII Rx clock rise time			2		ns	
t_{MRF}	MII Rx clock fall time			2		ns	
t_{MRH}	MII Rx clock HIGH time	100BASE-X	14			ns	
t_{MRL}	MII Rx clock LOW time	100BASE-X	14			ns	
t_{MRH}	MII Rx clock HIGH time	10BASE-X	140			ns	
t_{MRL}	MII Rx clock LOW time	10BASE-X	140			ns	
t_{MRP}	MII Rx clock period	100BASE-X		40.0		ns	± 100 PPM
t_{MRP}	MII Rx clock period	10BASE-X		400.0		ns	± 100 PPM
t_{MIOD}	MII Tx Output Delay	Positive edge of GMII Tx clock	0.0		25.0	ns	
t_{MIIS}	MII Rx Input Setup	Positive edge of GMII Tx clock	10.0			ns	
t_{MIIH}	MII Rx Input Hold	Positive edge of GMII Tx clock	10.0			ns	

9.4.6 Management Interface

The management interface is a serial interface that consists of a clock and a bidirectional data pin. The management interface is used to interface to the PHY devices. The MDC clock pin is generated only when the management interface performs a management cycle. Figure 9.6 shows the timing diagram and Table 9.9 lists the parameters of the Management Interface

Figure 9.6 Management Interface Timing Diagram

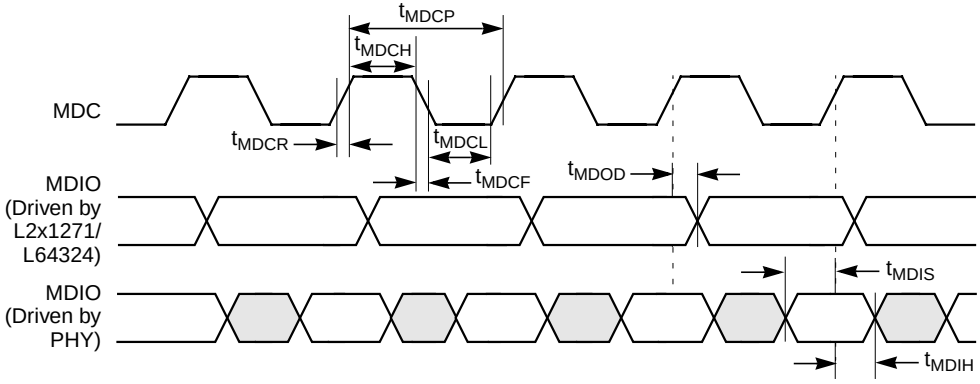


Table 9.9 Management Interface Timing

Sym	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{MDCR}	Management Data Clock (MDC) rise time			2		ns	
t_{MDCF}	Management Data Clock (MDC) fall time			2		ns	
t_{MDCH}	Management Data Clock (MDC) HIGH time		160			ns	
t_{MDCL}	Management Data Clock LOW time		160			ns	
t_{MDCP}	Management Data Clock (MDC) period		400			ns	

Table 9.9 Management Interface Timing (Cont.)

Sym	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{MDOD}	Management Data Output Delay (Driven by L2x1271/L64324)	Positive edge of MDC	10.0		300	ns	
t_{MDIS}	Management Data Input Setup (Driven by PHY)	Positive edge of MDC	100			ns	
t_{MDIH}	Management Data Input Hold (Driven by PHY)	Positive edge of MDC	0.0			ns	

9.4.7 SDRAM Interface

The L64324 contains a 96-bit wide SDRAM interface. The SDRAM interface outputs EXT_CLK[2:0], CSn, CASn, RASn, WEn, SDRAM_BS[1:0], SDRAM_DQMn[11:0] and ADDR[10:0]. The interface contains the bidirectional data bus DATA[95:0]. The positive edge of the clock on EXT_CLK[2:0] is used as timing reference for all SDRAM cycles. Figure 9.7 shows the timing diagram and Table 9.10 lists the parameters.of the SDRAM Interface.

Figure 9.7 SDRAM Interface Timing Diagram

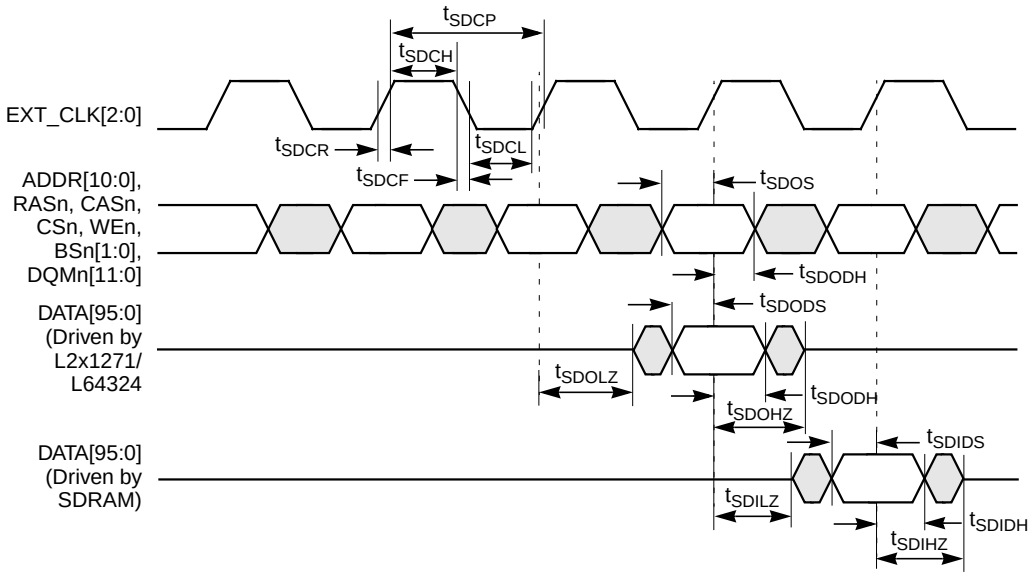


Table 9.10 SDRAM Interface Timing

Sym	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{SDCR}	SDRAM Clock rise time				1	ns	
t_{SDCF}	SDRAM Clock fall time				1	ns	
t_{SDCH}	SDRAM Clock HIGH time		3.5			ns	
t_{SDCL}	SDRAM Clock LOW time		3.5			ns	
t_{SDCP}	SDRAM Clock period			8.0		ns	± 100 PPM
t_{SDOS}	SDRAM Command Setup	Positive edge of SDRAM Clock	2.5			ns	
t_{SDOH}	SDRAM Command Hold	Positive edge of SDRAM Clock	1.5			ns	
t_{SDODS}	SDRAM Data Setup (driven by L2x1271/L64324)	Positive edge of SDRAM Clock	2.5			ns	
t_{SDODH}	SDRAM Data Hold (driven by L2x1271/L64324)	Positive edge of SDRAM Clock	1.5			ns	
t_{SDOLZ}	SDRAM Data output LOW Impedance (driven by L2x1271/L64324)	Positive edge of SDRAM Clock	3.0			ns	
t_{SDOHZ}	SDRAM Data output to HIGH Impedance (driven by L2x1271/L64324)	Positive edge of SDRAM Clock			5.0	ns	
t_{SDIDS}	SDRAM Data Setup (driven by SDRAM)	Positive edge of SDRAM Clock	1.5			ns	
t_{SDIDH}	SDRAM Data Hold (driven by SDRAM)	Positive edge of SDRAM Clock	1.0			ns	
t_{SDODS}	SDRAM Data output to LOW impedance (driven by SDRAM)	Positive edge of SDRAM Clock	1.0			ns	
t_{SDIHZ}	SDRAM Data Setup (driven by SDRAM)	Positive edge of SDRAM Clock			6.0	ns	

9.4.8 LED Interface

The L64324 contains a serial LED interface. The serial LED interface shifts out the LED data based on the LED clock. Externally a 74595 device is used to convert from serial to parallel, and drive the LEDs. The L64324 also generates a LED load pulse to be used by the 74595 device. The timings of this interface is shown in Figure 9.8 and Table 9.11.

Figure 9.8 LED Interface Timing Diagram

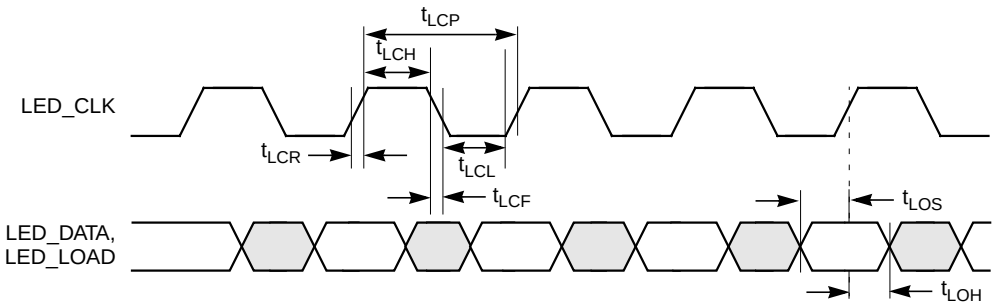


Table 9.11 LED Interface Timing

Sym	Parameter	Reference	Min	Typ	Max	Unit	Note
t_{LCR}	LED Clock rise time			3		ns	
t_{LCF}	LED Clock fall time			3		ns	
t_{LCH}	LED Clock HIGH time		80			ns	
t_{LCL}	LED Clock LOW time					ns	
t_{LCP}	LED Clock period		1.0		15.8	µs	
t_{LOS}	LED Outputs Setup		75.0			ns	
t_{LOH}	LED Outputs Hold		5.0			ns	

9.5 Pinout and Packaging

Table 9.12 is a L64324 388-Pin PBGA pinout listing the signal and the BGA pin. Figure 9.9 is a top view of the 388-pin BGA and Figure 9.10 is a mechanical drawing.

Table 9.12 L64324 388-Pin PBGA Pinout

Signal	Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal	Pin
ADDR[0]	B6	DATA[53]	B23	GND	D14	MAC_CLK	AD26	TXD[3]	AE7
ADDR[1]	A6	DATA[54]	A22	GND	D19	MDC	AD14	TXD[4]	AE9
ADDR[2]	C6	DATA[55]	B21	GND	D23	MDIO	AD15	TXD[5]	AD10
ADDR[3]	C18	DATA[56]	A21	GND	H4	RASn	A8	TXD[6]	AF12
ADDR[4]	A19	DATA[57]	C21	GND	J23	RBC[0]P1	M3	TXD[7]	AE3
ADDR[5]	B19	DATA[58]	D22	GND	L11	RBC[0]P2	AA3	TXD[8]	AF4
ADDR[6]	C19	DATA[59]	C22	GND	L12	RBC[1]P1	L3	TXD[9]	AE6
ADDR[7]	D20	DATA[60]	A23	GND	L13	RBC[1]P2	AA2	TXD[10]	AD8
ADDR[8]	B20	DATA[61]	A24	GND	L14	RESET_INn	AD25	TXD[11]	AF10
ADDR[9]	C20	DATA[62]	B24	GND	L15	RESET_OUTn	AC24	TXD[12]	AE11
ADDR[10]	B7	DATA[63]	A25	GND	L16	RESETP1	C2	TXD[13]	AF15
BSn[0]	C8	DATA[64]	A12	GND	M11	RESETP2	R1	TXD[14]	AF17
BSn[1]	A7	DATA[65]	B12	GND	M12	RI/ARM_TMS	AC22	TXD[15]	AE18
CASn	C9	DATA[66]/SS[0]	C12	GND	M13	RXCK/GRXCKP1	P4	TXD[16]	AF21
CFG_CLRn	AC15	DATA[67]/SS[1]	B11	GND	M14	RXCK/GRXCKP2	AD2	TXD[17]	AE22
CLK[0]	AF8	DATA[68]/FLASH_W[0]	C11	GND	M15	RXD[1]	AF3	TXD[18]	AF24
CLK[1]	AD12	DATA[69]/FLASH_W[1]	A10	GND	M16	RXD[2]	AE5	TXD[19]	AF14
CLK[2]	AF19	DATA[70]/FLASH_W[2]	B10	GND	N4	RXD[3]	AF6	TXD[20]	AE16
CLK[3]	AE24	DATA[71]/FLASH_W[3]	C10	GND	N11	RXD[4	AF9	TXD[21]	AD17
COLP1	N3	DATA[72]	B15	GND	N12	RXD[5]	AE10	TXD[22]	AE20
COLP2	AC3	DATA[73]	A15	GND	N13	RXD[6]	AD11	TXD[23]	AD21
CRSP1	M4	DATA[74]	D15	GND	N14	RXD[7]	AF2	TXD[24]	AF23
CRSP2	AB4	DATA[75]	A14	GND	N15	RXD[8]	AE4	TXD_D/RTS	AD7
CSn	B8	DATA[76]	C14	GND	N16	RXD[9]	AD6	TXDP1[0]	P1
CTS/SCAN_EN	AD18	DATA[77]	A13	GND	P11	RXD[10]	AE8	TXDP1[1]	M2
DATA[0]	T25	DATA[78]/HW_ID[0]	B13	GND	P12	RXD[11]	AD9	TXDP1[2]	M1
DATA[1]	T26	DATA[79]/HW_ID[1]	C13	GND	P13	RXD[12]	AF11	TXDP1[3]	L1
DATA[2]	R24	DATA[80]/B_ID[0]	A3	GND	P14	RXD[13]	AE14	TXDP1[4]	K2
DATA[3]	R25	DATA[81]/B_ID[1]	C4	GND	P15	RXD[14]	AD16	TXDP1[5]	K1
DATA[4]	R26	DATA[82]/B_ID[2]	B4	GND	P16	RXD[15]	AF18	TXDP1[6]	J2
DATA[5]	R23	DATA[83]/B_ID[3]	A4	GND	P23	RXD[16]	AD20	TXDP1[7]	J1
DATA[6]	P25	DATA[84]/HW_ID[2]	D5	GND	R11	RXD[17]	AF22	TXDP1[8]/TXENP1	H2
DATA[7]	P24	DATA[85]/HW_ID[[3]	C5	GND	R12	RXD[18]	AE23	TXDP1[9]/TXERP1	F1
DATA[8]/FADDR[0]	V24	DATA[86]/DEBUGn	A5	GND	R13	RXD[19]	AE13	TXDP2[0]	AD1
DATA[9]/FADDR[1]	V25	DATA[87]/BENCHn	B5	GND	R14	RXD[20]	AE15	TXDP2[1]	AC1
DATA[10]/FDDR[2]	V26	DATA[88]	B16	GND	R15	RXD[21]	AE17	TXDP2[2]	AB3
DATA[11]/FDDR[3]	U24	DATA[89]	C16	GND	R16	RXD[22]	AF20	TXDP2[3]	AB1
DATA[12]/FDDR[4]	U25	DATA[90]	D17	GND	T11	RXD[23]	AE21	TXDP2[4]	AA1
DATA[13]/FDDR[5]	U26	DATA[91]	A17	GND	T12	RXD[24]	AD22	TXDP2[5]	Y2
DATA[14]/FDDR[6]	U23	DATA[92]	B17	GND	T13	RXD_C	AC14	TXDP2[6]	Y1
DATA[15]FDDR[7]	T24	DATA[93]	C17	GND	T14	RXD_D/DSR	AC5	TXDP2[7]	W2
DATA[16]/FDDR[8]	L24	DATA[94]	D18	GND	T15	RXDP1[0]	B1	TXDP2[8]/TXENP2	W1
DATA[17]/FDDR[9]	L25	DATA[95]	A18	GND	T16	RXDP1[1]	C1	TXDP2[9]/TXERP2	U1
DATA[18]/FDDR[10]	L26	DCD/SCAN_IN	AC19	GND	V4	RXDP1[2]	E1	VDD-2.5V	A11
DATA[19]/FDDR[11]	M24	DEBUG_ACK	D13	GND	W23	RXDP1[3]	E2	VDD-2.5V	A20
DATA[20]/FDDR[12]	M25	DEBUG_REQ	D12	GND	AC4	RXDP1[4]	G1	VDD-2.5V	B14
DATA[21]/FDDR[13]	M26	DQMn[0]	N26	GND	AC8	RXDP1[5]	D3	VDD-2.5V	C7
DATA[22]/FDDR[14]	N24	DQMn[1]	W26	GND	AC13	RXDP1[6]	E3	VDD-2.5V	F3
DATA[23]/FDDR[15]	N25	DQMn[2]	K26	GND	AC18	RXDP1[7]	G3	VDD-2.5V	G23
DATA[24]/FDDR[16]	AB26	DQMn[3]	W25	GND	AC23	RXDP1[8]/RXDVP1	H3	VDD-2.5V	K24
DATA[25]/FDDR[17]	AA24	DQMn[4]	E25	GND	AD3	RXDP1[9]/RXERP1	K3	VDD-2.5V	N1
DATA[26]/FDDR[18]	AA25	DQMn[5]	E24	GND	AD24	RXDP2[0]	P2	VDD-2.5V	P26
DATA[27]/FDDR[19]	AA26	DQMn[6]	D25	GND	AE1	RXDP2[1]	P3	VDD-2.5V	T1
DATA[28]/FDDR[20]	Y24	DQMn[7]	E23	GND	AE2	RXDP2[2]	R3	VDD-2.5V	W3
DATA[29]/FDDR[21]	Y25	DQMn[8]	A9	GND	AE25	RXDP2[3]	T2	VDD-2.5V	Y23
DATA[30]/FDDR[22]	Y26	DQMn[9]	C15	GND	AF1	RXDP2[4]	U2	VDD-2.5V	AC20
DATA[31]	W24	DQMn[10]	B3	GND	AF25	RXDP2[5]	U3	VDD-2.5V	AD5
DATA[32]	K25	DQMn[11]	B18	GND	AF26	RXDP2[6]	U4	VDD-2.5V	AF13
DATA[33]	K23	DTR/SCAN_OUT	AD19	GTXCLKP1	L2	RXDP2[7]	V3	VDD-2.5V	AF16
DATA[34]	J26	DUMPn	G4	GTXCLKP2	AB2	RXDP2[8]/RXDVP2	W4	VDD-3.3V	D6
DATA[35]	J24	EXT_CLK[0]	AB25	IDP1[0]	D1	RXDP2[9]/RXERP2	Y3	VDD-3.3V	D11
DATA[36]	H25	EXT_CLK[1]	D26	IDP1[1]	G2	SPARE1	K4	VDD-3.3V	D16
DATA[37]	G25	EXT_CLK[2]	A16	IDP1[2]	J3	SPARE2	J4	VDD-3.3V	D21
DATA[38]	F25	FAN_FAULTn	V23	IDP1[3]	D2	SYNC/TXCKP1	F2	VDD-3.3V	F4
DATA[39]	E26	FLASH_CSn	AC26	IDP2[0]	R2	SYNC/TXCKP2	V2	VDD-3.3V	F23
DATA[40]	F24	FLASH_OEn	AC25	IDP2[1]	V1	SYNC[0]	AF7	VDD-3.3V	L4
DATA[41]	F26	FLASH_WEn	AB24	IDP2[2]	Y4	SYNC[1]	AE12	VDD-3.3V	L23
DATA[42]	G24	GLOBAL_TN	AB23	IDP2[3]	T3	SYNC[2]	AE19	VDD-3.3V	T4
DATA[43]	G26	GND	A1	IDDTN_ENABLE	N23	SYNC[3]	AD23	VDD-3.3V	T23
DATA[44]	H23	GND	A2	INTR	AD13	SYS_CLK	AE26	VDD-3.3V	AA4
DATA[45]	H24	GND	A26	INTRP1	H1	TCK	D8	VDD-3.3V	AA23
DATA[46]	H26	GND	B2	INTRP2	R4	TDI	E4	VDD-3.3V	AC6
DATA[47]	J25	GND	B25	LED_CLK	AC9	TDO	D10	VDD-3.3V	AC11
DATA[48]	C26	GND	B26	LED_DATA	AC12	TMS	D7	VDD-3.3V	AC16
DATA[49]	C25	GND	C3	LED_LOADn	AC10	TRST	M23	VDD-3.3V	AC21
DATA[50]	D24	GND	C24	LED_MODE_SWn	AC17	TXD_Cn	AC7	WEn	B9
DATA[51]	C23	GND	D4	LINK/SDnP1	N2	TXD[1]	AD4		
DATA[52]	B22	GND	D9	LINK/SDnP2	AC2	TXD[2]	AF5		

P1 = Uplink 1 and P2 = Uplink 2

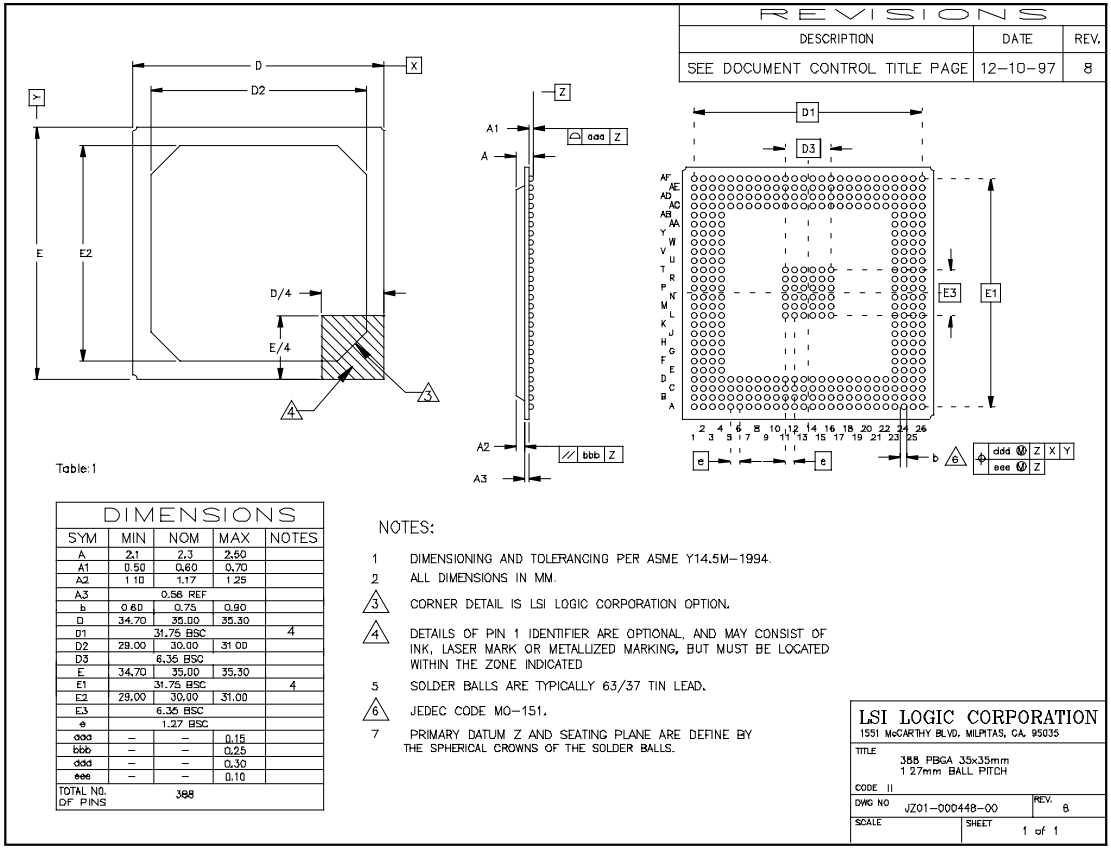
Figure 9.9 L64324 388-Pin PBGA Top View

A1 GND	A2 GND	A3 DATA[80]/ B_ID[0]	A4 DATA[83]/] B_ID[3]	A5 DATA[86]/ DEBUG	A6 ADDR[1]	A7 BSn[1]	A8 RASn	A9 DQMn[8]	A10 DATA[69]/ FLASH_W[1]	A11 VDD-2.5V	A12 DATA[64]	A13 DATA[77]
B1 RXDP1[.0]	B2 GND	B3 DQMn[10]	B4 DATA[82]/ B_ID[2]	B5 DATA[87]/ BENCHn	B6 ADDR[0]	B7 ADDR[10]	B8 CSn	B9 WEn	B10 DATA[70]/ FLASH_W[2]	B11 DATA[67]/ SS[1]	B12 DATA[65]	B13 DATA[78]/ HW_ID[0]
C1 RXDP1[1]	C2 RESETP1	C3 GND	C4 DATA[81]/ B_ID[1]	C5 DATA[85]/ HW_ID[3]	C6 ADDR[2]	C7 VDD-2.5V	C8 BSn[0]	C9 CASn	C10 DATA[71]/ FLASH_W[3]	C11 DATA[68]/ FLASH_W[0]	C12 DATA[66]/ SS[0]	C13 DATA[79]/ HW_ID[1]
D1 IDP1[0]	D2 IDP1[3]	D3 RXDP1[5]	D4 GND	D5 DATA[84]/ HW_ID[2]	D6 VDD-3.3V	D7 TMS	D8 TCK	D9 GND	D10 TDO	D11 VDD-3.3V	D12 DEBUG_REQ	D13 DEBUG_ACK
E1 RXDP1[2]	E2 RXDP1[3]	E3 RXDP1[6]	E4 TDI									
F1 TXDP1[9]/ TXERP1	F2 SYNC/ TXCKP1	F3 VDD-2.5V	F4 VDD-3.3V									
G1 RXDP1[4]	G2 IDP1[1]	G3 RXDP1[7]	G4 DUMP									
H1 INTRP1	H2 TXDP1[8]/ TXENP1	H3 RXDP1[8]/ RXDVP1	H4 GND									
J1 TXDP1[7]	J2 TXDP1[6]	J3 IDP1[2]	J4 SPARE2									
K1 TXDP1[5]	K2 TXDP1[4]	K3 RXDP1[9]/ RXERP1	K4 SPARE1									
L1 TXDP1[3]	L2 GTXCLKP1	L3 RBC[1]P1	L4 VDD-3.3V							L11 GND	L12 GND	L13 GND
M1 TXDP1[2]	M2 TXDP1[1]	M3 RBC[0]P1	M4 CRSP1							M11 GND	M12 GND	M13 GND
N1 VDD-2.5V	N2 LINK/ SDnP1	N3 COLP1	N4 GND							N11 GND	N12 GND	N13 GND
P1 TXDP1[0]	P2 RXDP2[0]	P3 RXDP2[1]	P4 RXCK/ GRXCKP1							P11 GND	P12 GND	P13 GND
R1 RESETP2	R2 IDP2[0]	R3 RXDP2[2]	R4 INTRP2							R11 GND	R12 GND	R13 GND
T1 VDD-2.5V	T2 RXDP2[3]	T3 IDP2[3]	T4 VDD-3.3V							T11 GND	T12 GND	T13 GND
U1 TXDP2[9]/ TXERP2	U2 RXDP2[4]	U3 RXDP2[5]	U4 RXDP2[6]									
V1 IDP2[1]	V2 SYNC/ TXCKP2	V3 RXDP2[7]	V4 GND									
W1 TXDP2[8]/ TXENP2	W2 TXDP2[7]	W3 VDD-2.5V	W4 RXDP2[8]/ RXDVP2									
Y1 TXDP2[6]	Y2 TXDP2[5]	Y3 RXDP2[9]/ RXERP2	Y4 IDP2[2]									
AA1 TXDP2[4]	AA2 RBC[1]P2	AA3 RBC[0]P2	AA4 VDD-3.3V									
AB1 TXDP2[3]	AB2 GTXCLKP2	AB3 TXDP2[2]	AB4 CRSP2									
AC1 TXDP2[1]	AC2 LINK_ SDnP2	AC3 COLP2	AC4 GND	AC5 RXD_D/DSR	AC6 VDD-3.3V	AC7 TXD_Cn	AC8 GND	AC9 LED_CLK	AC10 LED_LOADn	AC11 VDD-3.3V	AC12 LED_DATA	AC13 GND
AD1 TXDP2[0]	AD2 RXCK/ GRXCKP2	AD3 GND	AD4 TXD[1]	AD5 VDD-2.5V	AD6 RXD[9]	AD7 TXD_D/RTS	AD8 TXD[10]	AD9 RXD[11]	AD10 TXD[5]	AD11 RXD[6]	AD12 CLK[1]	AD13 INTR
AE1 GND	AE2 GND	AE3 TXD[7]	AE4 RXD[8]	AE5 RXD[2]	AE6 TXD[9]	AE7 TXD[3]	AE8 RXD[10]	AE9 TXD[4]	AE10 RXD[5]	AE11 TXD[12]	AE12 SYNC[1]	AE13 RXD[19]
AF1 GND	AF2 RXD[7]	AF3 RXD[1]	AF4 TXD[8]	AF5 TXD[2]	AF6 RXD[3]	AF7 SYNC[0]	AF8 CLK[0]	AF9 RXD[4]	AF10 TXD[11]	AF11 RXD[12]	AF12 TXD[6]	AF13 VDD-2.5V

Figure 9.9 L64324 388-Pin BGA Top View (Cont.)

A14 DATA[75]	A15 DATA[73]	A16 EXT_CLK[2]	A17 DATA[91]	A18 DATA[95]	A19 ADDR[4]	A20 VDD-2.5V	A21 DATA[56]	A22 DATA[54]	A23 DATA[60]	A24 DATA[61]	A25 DATA[63]	A26 GND
B14 VDD-2.5V	B15 DATA[72]	B16 DATA[88]	B17 DATA[92]	B18 DQMn[11]	B19 ADDR[5]	B20 ADDR[8]	B21 DATA[55]	B22 DATA[52]	B23 DATA[53]	B24 DATA[62]	B25 GND	B26 GND
C14 DATA[76]	C15 DQMn[9]	C16 DATA[89]	C17 DATA[93]	C18 ADDR[3]	C19 ADDR[6]	C20 ADDR[9]	C21 DATA[57]	C22 DATA[59]	C23 DATA[51]	C24 GND	C25 DATA[49]	C26 DATA[48]
D14 GND	D15 DATA[74]	D16 VDD-3.3V	D17 DATA[90]	D18 DATA[94]	D19 GND	D20 ADDR[7]	D21 VDD-3.3V	D22 DATA[58]	D23 GND	D24 DATA[50]	D25 DQMn[6]	D26 EXT_CLK[1]
									E23 DQMn[7]	E24 DQMn[5]	E25 DQMn[4]	E26 DATA[39]
									F23 VDD-3.3V	F24 DATA[40]	F25 DATA[38]	F26 DATA[41]
									G23 VDD-2.5V	G24 DATA[42]	G25 DATA[37]	G26 DATA[43]
									H23 DATA[44]	H24 DATA[45]	H25 DATA[36]	H26 DATA[46]
									J23 GND	J24 DATA[35]	J25 DATA[47]	J26 DATA[34]
									K23 DATA[33]	K24 VDD-2.5V	K25 DATA[32]	K26 DQMn[2]
L14 GND	L15 GND	L16 GND							L23 VDD-3.3V	L24 DATA[16]/ FDDR[8]	L25 DATA[17]/ FDDR[9]	L26 DATA[18]/ FDDR[10]
M14 GND	M15 GND	M16 GND							M23 TRST	M24 DATA[19]/ FDDR[11]	M25 DATA[20]/ FDDR[12]	M26 DATA[21]/ FDDR[13]
N14 GND	N15 GND	N16 GND							N23 IDDTN_ ENABLE	N24 DATA[22]/ FDDR[14]	N25 DATA[23]/ FDDR[15]	N26 DQMn[0]
P14 GND	P15 GND	P16 GND							P23 GND	P24 DATA[7]	P25 DATA[6]	P26 VDD-2.5V
R14 GND	R15 GND	R16 GND							R23 DATA[5]	R24 DATA[2]	R25 DATA[3]	R26 DATA[4]
T14 GND	T15 GND	T16 GND							T23 VDD-3.3V	T24 DATA[15]/ FDDR[7]	T25 DATA[0]	T26 DATA[1]
									U23 DATA[14]/ FDDR[6]	U24 DATA[11]/ FDDR[3]	U25 DATA[12]/ FDDR[4]	U26 DATA[13]/ FDDR[5]
									V23 FAN_FAULTn	V24 DATA[8]/ FADDR[0]	V25 DATA[9]/ FADDR[1]	V26 DATA[10]/ FDDR[2]
									W23 GND	W24 DATA[31]	W25 DQMn[3]	W26 DQMn[1]
									Y23 VDD-2.5V	Y24 DATA[28]/ FDDR[20]	Y25 DATA[29]/ FDDR[21]	Y26 DATA[30]/ FDDR[22]
									AA23 VDD-3.3V	AA24 DATA[25]/ FDDR[17]	AA25 DATA[26]/ FDDR[18]	AA26 DATA[27]/ FDDR[19]
									AB23 GLOBAL_TN	AB24 FLASH_WEn	AB25 EXT_CLK[0]	AB26 DATA[24]/ FDDR[16]
AC14 RXD_C	AC15 CFG_CLRn	AC16 VDD-3.3V	AC17 LED_MODE_ SWn	AC18 GND	AC19 DCD/ SCAN_IN	AC20 VDD-2.5V	AC21 VDD-3.3V	AC22 RI/ARM_TMS	AC23 GND	AC24 RESET_OUTn	AC25 FLASH_OEn	AC26 FLASH_CSn
AD14 MDC	AD15 MDIO	AD16 RXD[14]	AD17 TXD[21]	AD18 CTS/ SCAN_EN	AD19 DTR/ SCAN_OUT	AD20 RXD[16]	AD21 TXD[23]	AD22 RXD[24]	AD23 SYNC[3]	AD24 GND	AD25 RESET_INn	AD26 MAC_CLK
AE14 RXD[13]	AE15 RXD[20]	AE16 TXD[20]	AE17 RXD[21]	AE18 TXD[15]	AE19 SYNC[2]	AE20 TXD[22]	AE21 RXD[23]	AE22 TXD[17]	AE23 RXD[18]	AE24 CLK[3]	AE25 GND	AE26 SYS_CLK
AF14 TXD[19]	AF15 TXD[13]	AF16 VDD-2.5V	AF17 TXD[14]	AF18 RXD[15]	AF19 CLK[2]	AF20 RXD[22]	AF21 TXD[16]	AF22 RXD[17]	AF23 TXD[24]	AF24 TXD[18]	AF25 GND	AF26 GND

Figure 9.10 388-Pin BGA (II) Mechanical Drawing



Important: **This drawing may not be the latest version. For board layout and manufacturing, obtain the most recent engineering drawings from your LSI Logic marketing representative by requesting the outline drawing for package code II.**

Customer Feedback

We would appreciate your feedback on this document. Please copy the following page, add your comments, and fax it to us at the number shown.

If appropriate, please also fax copies of any marked-up pages from this document.

Important: Please include your name, phone number, fax number, and company address so that we may contact you directly for clarification or additional information.

Thank you for your help in improving the quality of our documents.

Reader's Comments

Fax your comments to: LSI Logic Corporation
Technical Publications
M/S E-198
Fax: 408.433.4333

Please tell us how you rate this document: *L64324 24 x 2 Fast Ethernet Intelligent Switch.* Place a check mark in the appropriate blank for each category.

	Excellent	Good	Average	Fair	Poor
Completeness of information	____	____	____	____	____
Clarity of information	____	____	____	____	____
Ease of finding information	____	____	____	____	____
Technical content	____	____	____	____	____
Usefulness of examples and illustrations	____	____	____	____	____
Overall manual	____	____	____	____	____

What could we do to improve this document?

_____

_____

_____

_____

If you found errors in this document, please specify the error and page number. If appropriate, please fax a marked-up copy of the page(s).

_____

_____

_____

Please complete the information below so that we may contact you directly for clarification or additional information.

Name _____ Date _____

Telephone _____ Fax _____

Title _____

Department _____ Mail Stop _____

Company Name _____

Street _____

City, State, Zip _____

U.S. Distributors by State

A. E. Avnet Electronics
http://www.hh.avnet.com
B. M. Bell Microproducts, Inc. (for HAB's)
http://www.bellmicro.com
I. E. Insight Electronics
http://www.insight-electronics.com
W. E. Wyle Electronics
http://www.wyle.com

Alabama
Daphne
I. E. Tel: 334.626.6190
Huntsville
A. E. Tel: 256.837.8700
B. M. Tel: 256.705.3559
I. E. Tel: 256.830.1222
W. E. Tel: 800.964.9953

Alaska
A. E. Tel: 800.332.8638

Arizona
Phoenix
A. E. Tel: 480.736.7000
B. M. Tel: 602.267.9551
W. E. Tel: 800.528.4040
Tempe
I. E. Tel: 480.829.1800
Tucson
A. E. Tel: 520.742.0515

Arkansas
W. E. Tel: 972.235.9953

California
Agoura Hills
B. M. Tel: 818.865.0266
Granite Bay
B. M. Tel: 916.523.7047
Irvine
A. E. Tel: 949.789.4100
B. M. Tel: 949.470.2900
I. E. Tel: 949.727.3291
W. E. Tel: 800.626.9953
Los Angeles
A. E. Tel: 818.594.0404
W. E. Tel: 800.288.9953
Sacramento
A. E. Tel: 916.632.4500
W. E. Tel: 800.627.9953
San Diego
A. E. Tel: 858.385.7500
B. M. Tel: 858.597.3010
I. E. Tel: 800.677.6011
W. E. Tel: 800.829.9953
San Jose
A. E. Tel: 408.435.3500
B. M. Tel: 408.436.0881
I. E. Tel: 408.952.7000
Santa Clara
W. E. Tel: 800.866.9953
Woodland Hills
A. E. Tel: 818.594.0404
Westlake Village
I. E. Tel: 818.707.2101

Colorado
Denver
A. E. Tel: 303.790.1662
B. M. Tel: 303.846.3065
W. E. Tel: 800.933.9953
Englewood
I. E. Tel: 303.649.1800
Idaho Springs
B. M. Tel: 303.567.0703

Connecticut
Cheshire
A. E. Tel: 203.271.5700
I. E. Tel: 203.272.5843
Wallingford
W. E. Tel: 800.605.9953

Delaware
North/South
A. E. Tel: 800.526.4812
Tel: 800.638.5988
B. M. Tel: 302.328.8968
W. E. Tel: 856.439.9110

Florida
Altamonte Springs
B. M. Tel: 407.682.1199
I. E. Tel: 407.834.6310
Boca Raton
I. E. Tel: 561.997.2540
Bonita Springs
B. M. Tel: 941.498.6011
Clearwater
I. E. Tel: 727.524.8850
Fort Lauderdale
A. E. Tel: 954.484.5482
W. E. Tel: 800.568.9953
Miami
B. M. Tel: 305.477.6406
Orlando
A. E. Tel: 407.657.3300
W. E. Tel: 407.740.7450
Tampa
W. E. Tel: 800.395.9953
St. Petersburg
A. E. Tel: 727.507.5000

Georgia
Atlanta
A. E. Tel: 770.623.4400
B. M. Tel: 770.980.4922
W. E. Tel: 800.876.9953
Duluth
I. E. Tel: 678.584.0812

Hawaii
A. E. Tel: 800.851.2282

Idaho
A. E. Tel: 801.365.3800
W. E. Tel: 801.974.9953

Illinois
North/South
A. E. Tel: 847.797.7300
Tel: 314.291.5350
Chicago
B. M. Tel: 847.413.8530
W. E. Tel: 800.853.9953
Schaumburg
I. E. Tel: 847.885.9700

Indiana
Fort Wayne
I. E. Tel: 219.436.4250
W. E. Tel: 888.358.9953
Indianapolis
A. E. Tel: 317.575.3500

Iowa
W. E. Tel: 612.853.2280
Cedar Rapids
A. E. Tel: 319.393.0033

Kansas
W. E. Tel: 303.457.9953
Kansas City
A. E. Tel: 913.663.7900
Lenexa
I. E. Tel: 913.492.0408

Kentucky
W. E. Tel: 937.436.9953
Central/Northern/ Western
A. E. Tel: 800.984.9503
Tel: 800.767.0329
Tel: 800.829.0146

Louisiana
W. E. Tel: 713.854.9953
North/South
A. E. Tel: 800.231.0253
Tel: 800.231.5775

Maine
A. E. Tel: 800.272.9255
W. E. Tel: 781.271.9953

Maryland
Baltimore
A. E. Tel: 410.720.3400
W. E. Tel: 800.863.9953
Columbia
B. M. Tel: 800.673.7461
I. E. Tel: 410.381.3131

Massachusetts
Boston
A. E. Tel: 978.532.9808
W. E. Tel: 800.444.9953
Burlington
I. E. Tel: 781.270.9400
Marlborough
B. M. Tel: 800.673.7459
Woburn
B. M. Tel: 800.552.4305

Michigan
Brighton
I. E. Tel: 810.229.7710
Detroit
A. E. Tel: 734.416.5800
W. E. Tel: 888.318.9953
Clarkston
B. M. Tel: 877.922.9363

Minnesota
Champlin
B. M. Tel: 800.557.2566
Eden Prairie
B. M. Tel: 800.255.1469
Minneapolis
A. E. Tel: 612.346.3000
W. E. Tel: 800.860.9953
St. Louis Park
I. E. Tel: 612.525.9999

Mississippi
A. E. Tel: 800.633.2918
W. E. Tel: 256.830.1119

Missouri
W. E. Tel: 630.620.0969
St. Louis
A. E. Tel: 314.291.5350
I. E. Tel: 314.872.2182

Montana
A. E. Tel: 800.526.1741
W. E. Tel: 801.974.9953

Nebraska
A. E. Tel: 800.332.4375
W. E. Tel: 303.457.9953

Nevada
Las Vegas
A. E. Tel: 800.528.8471
W. E. Tel: 702.765.7117

New Hampshire
A. E. Tel: 800.272.9255
W. E. Tel: 781.271.9953

New Jersey
North/South
A. E. Tel: 201.515.1641
Tel: 609.222.6400
Mt. Laurel
I. E. Tel: 856.222.9566
Pine Brook
B. M. Tel: 973.244.9668
W. E. Tel: 800.862.9953
Parsippany
I. E. Tel: 973.299.4425
Wayne
W. E. Tel: 973.237.9010

New Mexico
W. E. Tel: 480.804.7000
Albuquerque
A. E. Tel: 505.293.5119

U.S. Distributors by State (Continued)

New York
Hauppauge
I. E. Tel: 516.761.0960
Long Island
A. E. Tel: 516.434.7400
W. E. Tel: 800.861.9953
Rochester
A. E. Tel: 716.475.9130
I. E. Tel: 716.242.7790
W. E. Tel: 800.319.9953
Smithtown
B. M. Tel: 800.543.2008
Syracuse
A. E. Tel: 315.449.4927

North Carolina
Raleigh
A. E. Tel: 919.859.9159
I. E. Tel: 919.873.9922
W. E. Tel: 800.560.9953

North Dakota
A. E. Tel: 800.829.0116
W. E. Tel: 612.853.2280

Ohio
Cleveland
A. E. Tel: 216.498.1100
W. E. Tel: 800.763.9953
Dayton
A. E. Tel: 614.888.3313
I. E. Tel: 937.253.7501
W. E. Tel: 800.575.9953
Strongsville
B. M. Tel: 440.238.0404
Valley View
I. E. Tel: 216.520.4333

Oklahoma
W. E. Tel: 972.235.9953
Tulsa
A. E. Tel: 918.459.6000
I. E. Tel: 918.665.4664

Oregon
Beaverton
B. M. Tel: 503.524.1075
I. E. Tel: 503.644.3300
Portland
A. E. Tel: 503.526.6200
W. E. Tel: 800.879.9953

Pennsylvania
Mercer
I. E. Tel: 412.662.2707
Philadelphia
A. E. Tel: 800.526.4812
B. M. Tel: 877.351.2355
W. E. Tel: 800.871.9953
Pittsburgh
A. E. Tel: 412.281.4150
W. E. Tel: 440.248.9996

Rhode Island
A. E. 800.272.9255
W. E. Tel: 781.271.9953

South Carolina
A. E. Tel: 919.872.0712
W. E. Tel: 919.469.1502

South Dakota
A. E. Tel: 800.829.0116
W. E. Tel: 612.853.2280

Tennessee
W. E. Tel: 256.830.1119
East/West
A. E. Tel: 800.241.8182
Tel: 800.633.2918

Texas
Arlington
B. M. Tel: 817.417.5993
Austin
A. E. Tel: 512.219.3700
B. M. Tel: 512.258.0725
I. E. Tel: 512.719.3090
W. E. Tel: 800.365.9953
Dallas
A. E. Tel: 214.553.4300
B. M. Tel: 972.783.4191
W. E. Tel: 800.955.9953
El Paso
A. E. Tel: 800.526.9238
Houston
A. E. Tel: 713.781.6100
B. M. Tel: 713.917.0663
W. E. Tel: 800.888.9953
Richardson
I. E. Tel: 972.783.0800
Rio Grande Valley
A. E. Tel: 210.412.2047
Stafford
I. E. Tel: 281.277.8200

Utah
Centerville
B. M. Tel: 801.295.3900
Murray
I. E. Tel: 801.288.9001
Salt Lake City
A. E. Tel: 801.365.3800
W. E. Tel: 800.477.9953

Vermont
A. E. Tel: 800.272.9255
W. E. Tel: 716.334.5970

Virginia
A. E. Tel: 800.638.5988
W. E. Tel: 301.604.8488
Haymarket
B. M. Tel: 703.754.3399
Springfield
B. M. Tel: 703.644.9045

Washington
Kirkland
I. E. Tel: 425.820.8100
Maple Valley
B. M. Tel: 206.223.0080
Seattle
A. E. Tel: 425.882.7000
W. E. Tel: 800.248.9953

West Virginia
A. E. Tel: 800.638.5988

Wisconsin
Milwaukee
A. E. Tel: 414.513.1500
W. E. Tel: 800.867.9953
Wauwatosa
I. E. Tel: 414.258.5338

Wyoming
A. E. Tel: 800.332.9326
W. E. Tel: 801.974.9953

Sales Offices and Design Resource Centers

LSI Logic Corporation
Corporate Headquarters
1551 McCarthy Blvd
Milpitas CA 95035
Tel: 408.433.8000
Fax: 408.433.8989

NORTH AMERICA

California
Irvine
18301 Von Karman Ave
Suite 900
Irvine, CA 92612
♦ Tel: 949.809.4600
Fax: 949.809.4444

Pleasanton Design Center
5050 Hopyard Road, 3rd Floor
Suite 300
Pleasanton, CA 94588
Tel: 925.730.8800
Fax: 925.730.8700

San Diego
7585 Ronson Road
Suite 100
San Diego, CA 92111
Tel: 858.467.6981
Fax: 858.496.0548

Silicon Valley
1551 McCarthy Blvd
Sales Office
M/S C-500
Milpitas, CA 95035
♦ Tel: 408.433.8000
Fax: 408.954.3353
Design Center
M/S C-410
Tel: 408.433.8000
Fax: 408.433.7695

Wireless Design Center
11452 El Camino Real
Suite 210
San Diego, CA 92130
Tel: 858.350.5560
Fax: 858.350.0171

Colorado
Boulder
4940 Pearl East Circle
Suite 201
Boulder, CO 80301
♦ Tel: 303.447.3800
Fax: 303.541.0641

Colorado Springs
4420 Arrowswest Drive
Colorado Springs, CO 80907
Tel: 719.533.7000
Fax: 719.533.7020

Fort Collins
2001 Danfield Court
Fort Collins, CO 80525
Tel: 970.223.5100
Fax: 970.206.5549

Florida
Boca Raton
2255 Glades Road
Suite 324A
Boca Raton, FL 33431
Tel: 561.989.3236
Fax: 561.989.3237

Georgia
Alpharetta
2475 North Winds Parkway
Suite 200
Alpharetta, GA 30004
Tel: 770.753.6146
Fax: 770.753.6147

Illinois
Oakbrook Terrace
Two Mid American Plaza
Suite 800
Oakbrook Terrace, IL 60181
Tel: 630.954.2234
Fax: 630.954.2235

Kentucky
Bowling Green
1262 Chestnut Street
Bowling Green, KY 42101
Tel: 270.793.0010
Fax: 270.793.0040

Maryland
Bethesda
6903 Rockledge Drive
Suite 230
Bethesda, MD 20817
Tel: 301.897.5800
Fax: 301.897.8389

Massachusetts
Waltham
200 West Street
Waltham, MA 02451
♦ Tel: 781.890.0180
Fax: 781.890.6158

Burlington - Mint Technology
77 South Bedford Street
Burlington, MA 01803
Tel: 781.685.3800
Fax: 781.685.3801

Minnesota
Minneapolis
8300 Norman Center Drive
Suite 730
Minneapolis, MN 55437
♦ Tel: 612.921.8300
Fax: 612.921.8399

New Jersey
Red Bank
125 Half Mile Road
Suite 200
Red Bank, NJ 07701
Tel: 732.933.2656
Fax: 732.933.2643

Cherry Hill - Mint Technology
215 Longstone Drive
Cherry Hill, NJ 08003
Tel: 856.489.5530
Fax: 856.489.5531

New York
Fairport
550 Willowbrook Office Park
Fairport, NY 14450
Tel: 716.218.0020
Fax: 716.218.9010

North Carolina
Raleigh
Phase II
4601 Six Forks Road
Suite 528
Raleigh, NC 27609
Tel: 919.785.4520
Fax: 919.783.8909

Oregon
Beaverton
15455 NW Greenbrier Parkway
Suite 235
Beaverton, OR 97006
Tel: 503.645.0589
Fax: 503.645.6612

Texas
Austin
9020 Capital of TX Highway North
Building 1
Suite 150
Austin, TX 78759
Tel: 512.388.7294
Fax: 512.388.4171

Plano
500 North Central Expressway
Suite 440
Plano, TX 75074
♦ Tel: 972.244.5000
Fax: 972.244.5001

Houston
20405 State Highway 249
Suite 450
Houston, TX 77070
Tel: 281.379.7800
Fax: 281.379.7818

Canada
Ontario
Ottawa
260 Hearst Way
Suite 400
Kanata, ON K2L 3H1
♦ Tel: 613.592.1263
Fax: 613.592.3253

INTERNATIONAL

France
Paris
LSI Logic S.A.
Immeuble Europa
53 bis Avenue de l'Europe
B.P. 139
78148 Velizy-Villacoublay
Cedex, Paris
♦ Tel: 33.1.34.63.13.13
Fax: 33.1.34.63.13.19

Germany
Munich
LSI Logic GmbH
Orleansstrasse 4
81669 Munich
♦ Tel: 49.89.4.58.33.0
Fax: 49.89.4.58.33.108

Stuttgart
Mittlerer Pfad 4
D-70499 Stuttgart
♦ Tel: 49.711.13.96.90
Fax: 49.711.86.61.428

Italy
Milan
LSI Logic S.P.A.
Centro Direzionale Colleoni Palazzo
Orione Ingresso 1
20041 Agrate Brianza, Milano
♦ Tel: 39.039.687371
Fax: 39.039.6057867

Japan
Tokyo
LSI Logic K.K.
Rivage-Shinagawa Bldg. 14F
4-1-8 Kounan
Minato-ku, Tokyo 108-0075
♦ Tel: 81.3.5463.7821
Fax: 81.3.5463.7820

Osaka
Crystal Tower 14F
1-2-27 Shiromi
Chuo-ku, Osaka 540-6014
♦ Tel: 81.6.947.5281
Fax: 81.6.947.5287

Sales Offices and Design Resource Centers (Continued)

Korea
Seoul
LSI Logic Corporation of Korea Ltd
10th Fl., Haesung 1 Bldg.
942, Daechi-dong,
Kangnam-ku, Seoul, 135-283
Tel: 82.2.528.3400
Fax: 82.2.528.2250

The Netherlands
Eindhoven
LSI Logic Europe Ltd
World Trade Center Eindhoven
Building 'Rijder'
Bogert 26
5612 LZ Eindhoven
Tel: 31.40.265.3580
Fax: 31.40.296.2109

Singapore
Singapore
LSI Logic Pte Ltd
7 Temasek Boulevard
#28-02 Suntec Tower One
Singapore 038987
Tel: 65.334.9061
Fax: 65.334.4749

Sweden
Stockholm
LSI Logic AB
Finlandsgatan 14
164 74 Kista
♦ Tel: 46.8.444.15.00
Fax: 46.8.750.66.47

Taiwan
Taipei
LSI Logic Asia, Inc.
Taiwan Branch
10/F 156 Min Sheng E. Road
Section 3
Taipei, Taiwan R.O.C.
Tel: 886.2.2718.7828
Fax: 886.2.2718.8869

United Kingdom
Bracknell
LSI Logic Europe Ltd
Greenwood House
London Road
Bracknell, Berkshire RG12 2UB
♦ Tel: 44.1344.426544
Fax: 44.1344.481039

♦ Sales Offices with Design Resource Centers

International Distributors

Australia
New South Wales
Reptechnic Pty Ltd
3/36 Bydown Street
Neutral Bay, NSW 2089
♦ Tel: 612.9953.9844
Fax: 612.9953.9683

Belgium
Acal nv/sa
Lozenberg 4
1932 Zaventem
Tel: 32.2.7205983
Fax: 32.2.7251014

China
Beijing
LSI Logic International Services Inc.
Beijing Representative Office
Room 708
Canway Building
66 Nan Li Shi Lu
Xicheng District
Beijing 100045, China
Tel: 86.10.6804.2534 to 38
Fax: 86.10.6804.2521

France
Rungis Cedex
Azzurri Technology France
22 Rue Saarinen
Sillic 274
94578 Rungis Cedex
Tel: 33.1.41806310
Fax: 33.1.41730340

Germany
Haar
EBV Elektronik
Hans-Pinsel Str. 4
D-85540 Haar
Tel: 49.89.4600980
Fax: 49.89.46009840

Munich
Avnet Emg GmbH
Stahlgruberring 12
81829 Munich
Tel: 49.89.45110102
Fax: 49.89.42.27.75

Wuennenberg-Haaren
Peacock AG
Graf-Zepplin-Str 14
D-33181 Wuennenberg-Haaren
Tel: 49.2957.79.1692
Fax: 49.2957.79.9341

Hong Kong
Hong Kong
AVT Industrial Ltd
Unit 608 Tower 1
Cheung Sha Wan Plaza
833 Cheung Sha Wan Road
Kowloon, Hong Kong
Tel: 852.2428.0008
Fax: 852.2401.2105

Serial System (HK) Ltd
2301 Nanyang Plaza
57 Hung To Road, Kwun Tong
Kowloon, Hong Kong
Tel: 852.2995.7538
Fax: 852.2950.0386

India
Bangalore
Spike Technologies India Private Ltd
951, Vijayalakshmi Complex,
2nd Floor, 24th Main,
J P Nagar II Phase,
Bangalore, India 560078
♦ Tel: 91.80.664.5530
Fax: 91.80.664.9748

Israel
Tel Aviv
Eastronics Ltd
11 Rozanis Street
P.O. Box 39300
Tel Aviv 61392
Tel: 972.3.6458777
Fax: 972.3.6458666

Japan
Tokyo
Daito Electron
Sogo Kojimachi No.3 Bldg
1-6 Kojimachi
Chiyoda-ku, Tokyo 102-8730
Tel: 81.3.3264.0326
Fax: 81.3.3261.3984

Global Electronics Corporation
Nichibei Time24 Bldg. 35 Tansu-cho
Shinjuku-ku, Tokyo 162-0833
Tel: 81.3.3260.1411
Fax: 81.3.3260.7100
Technical Center
Tel: 81.471.43.8200

Marubeni Solutions
1-26-20 Higashi
Shibuya-ku, Tokyo 150-0001
Tel: 81.3.5778.8662
Fax: 81.3.5778.8669

Shinki Electronics
Myuru Daikanyama 3F
3-7-3 Ebisu Minami
Shibuya-ku, Tokyo 150-0022
Tel: 81.3.3760.3110
Fax: 81.3.3760.3101

Yokohama-City
Innotech
2-15-10 Shin Yokohama
Kohoku-ku
Yokohama-City, 222-8580
Tel: 81.45.474.9037
Fax: 81.45.474.9065

Macnica Corporation
Hakusan High-Tech Park
1-22-2 Hadusan, Midori-Ku,
Yokohama-City, 226-8505
Tel: 81.45.939.6140
Fax: 81.45.939.6141

The Netherlands
Eindhoven
Acal Nederland b.v.
Beatrix de Rijkweg 8
5657 EG Eindhoven
Tel: 31.40.2.502602
Fax: 31.40.2.510255

Switzerland
Brugg
LSI Logic Sulzer AG
Mattenstrasse 6a
CH 2555 Brugg
Tel: 41.32.3743232
Fax: 41.32.3743233

Taiwan
Taipei
Avnet-Mercuries Corporation, Ltd
14F, No. 145,
Sec. 2, Chien Kuo N. Road
Taipei, Taiwan, R.O.C.
Tel: 886.2.2516.7303
Fax: 886.2.2505.7391

Lumax International Corporation, Ltd
7th Fl., 52, Sec. 3
Nan-Kang Road
Taipei, Taiwan, R.O.C.
Tel: 886.2.2788.3656
Fax: 886.2.2788.3568

Prospect Technology Corporation, Ltd
4Fl., No. 34, Chu Luen Street
Taipei, Taiwan, R.O.C.
Tel: 886.2.2721.9533
Fax: 886.2.2773.3756

Wintech Microeletronics Co., Ltd
7F., No. 34, Sec. 3, Pateh Road
Taipei, Taiwan, R.O.C.
Tel: 886.2.2579.5858
Fax: 886.2.2570.3123

United Kingdom
Maidenhead
Azzurri Technology Ltd
16 Grove Park Business Estate
Waltham Road
White Waltham
Maidenhead, Berkshire SL6 3LW
Tel: 44.1628.826826
Fax: 44.1628.829730

Milton Keynes
Ingram Micro (UK) Ltd
Garamonde Drive
Wymbush
Milton Keynes
Buckinghamshire MK8 8DF
Tel: 44.1908.260422

Swindon
EBV Elektronik
12 Interface Business Park
Bincknoll Lane
Wootton Bassett,
Swindon, Wiltshire SN4 8SY
Tel: 44.1793.849933
Fax: 44.1793.859555

♦ Sales Offices with Design Resource Centers